MMLF: Multi-Metric Latent Feature Analysis for High-Dimensional and Incomplete Data

Di Wu, *Member, IEEE,* Peng Zhang, Yi He, *Member, IEEE*, and Xin Luo, *Senior Member, IEEE*

Abstract—High-dimensional and incomplete (HDI) data are omnipresent in a variety of big data-related applications. Latent feature analysis (LFA) is a typical representation learning method that can extract useful yet latent knowledge from HDI data via low-rank embedding. Existing LFA-based models mostly adopt a single-metric-based modeling strategy, where the representation designed for the embedding *Loss* function is fixed and exclusive. However, real-world HDI data are commonly heterogeneous and have large diverse underlying patterns, making a single-metric-based model cannot represent such HDI data in a comprehensive and unbiased fashion. Motivated by this discovery, this paper proposes a multi-metric latent feature (MMLF) model whose ideas are two-fold: 1) two vector spaces and three L_p-norms are simultaneously adopted to develop six LFA variants, each of which possesses a unique merit, and 2) all the variants are aggregated with a tailored, self-adaptive weighting strategy. As such, the proposed MMLF enjoys the merits originated from a set of disparate metric spaces all at once, achieving the comprehensive and unbiased representation of HDI data. Theoretical study guarantees that MMLF attains evident performance gain. Extensive experiments on ten real-world HDI matrices, spanning a wide range of industrial and scientific areas, verify that the proposed MMLF significantly outperforms nine state-of-the-art, shallow and deep counterparts.

Index Terms—Online services, High-dimensional and Incomplete Data, Latent Feature Analysis, Representation Learning, Representation Metric, Multi-metric Modeling, Missing Data Estimation, .

1 Introduction

Matrices are a norm to describe the pairwise relationships among entities. For example, the user-item matrix has been widely adopted in profiling an online e-commerce service system [1], with its row and column vectors associating with users and items, respectively. Its each entry indicates the preference or interest of a user towards an item. Indeed, matrices abound in many science and industrial applications, influencing practically every aspect of our existence, such as Web service [1, 2], recommender systems [3], social service networks [4], sensing networks of mobile service [5], etc.

The crux of analyzing these data matrices yielded from large systems lies in their high-dimensional and incomplete (HDI) characteristics [6, 7]. This is conceptually and practically tangible, e.g., the numbers of users and items are unmanageably large in nowadays e-commerce platforms such as Amazon or eBay, while the observed entries (user-item interactions) of such large-scaled matrices are relatively very few [1, 8]. As such, the HDI matrices leave the values of most entries unknown, behind which rich and invaluable knowledge exists (e.g., users' potential preferences on items)[9, 10]. Hence, it is significant to learn accurate representations (i.e., latent features) from the HDI matrices for hidden knowledge extraction [11].

Latent feature analysis (LFA) is one of the most popular representation learning methods tailored for HDI matrices, thanks to its high accuracy, computational efficiency, and ease of scalability [9, 10]. An LFA model employs two low-rank latent feature matrices to produce an approximation of the original HDI matrix. The two matrices are iteratively trained by minimizing the approximation *Loss* gauged between the originally observed and estimated entries [12]. The better the observed entries are estimated, the more accurate latent features are extracted, where the two matrices represent the extracted latent features *w.r.t.* row entities and column entities of the original HDI matrix, respectively [6].

Despite the successes of existing LFA-based models, they share an essence that their representation learning strategy designed for the approximation *Loss* function is exclusive and fixed in a priori knowledge [6]. This leads to a single-metric representation of a target HDI matrix. However, the real-world HDI matrices are commonly heterogeneous, inclusive, and have large diverse underlying patterns, manifesting the limitations of such a single, fixed, and exclusive representation strategy [6, 13, 14]. An example is given to demonstrate these limitations in representing an HDI matrix as follows.

Example 1. As shown in the middle panel of Fig.1, given an HDI H with four row entities m_1, m_2, m_3, and m_4, and two column entities n_1 and n_2, the relationship between a row entity m and a column entity n is represented as an entry $h_{m,n}$, e.g., $h_{1,1}$=2. H shows that both m_1 and m_2 are similar to m_3 as $h_{1,1}$= $h_{3,1}$=2 and $h_{2,2}$= $h_{3,2}$=2, which deduces that m_1, m_2, and m_3 are all similar. The similarity among m_1, m_2, and m_3 is defined as '*global similarity*' because their similarity is deduced from all the observed entries of H. Besides, it is easy to find that n_1 and n_2 are similar as $h_{3,1}$= $h_{3,2}$=2. The similarity between n_1 and n_2 is defined as '*local similarity*' because it is directly discovered from the local observed entries of m_3 only. Another illustrative example is shown in Fig. 2 to detail the '*global/local similarity*'. Notably, m_4 is defined as an '*outlier*' because its relationships with both n_1 and n_2 are 100, which is much different from m_1, m_2, and m_3. To

- *D. Wu and X. Luo are with the College of Computer and Information Science, Southwest University, Chongqing 400715, China (e-mail: wudi.cigit@gmail.com, luoxin21@gmail.com).*
- *P. Zhang is with the School of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: s190231042@stu.cqupt.edu.cn)*
- *Y. He is with the Old Dominion University, Norfolk, Virginia 23462, USA. (e-mail: yihe@cs.odu.edu).*
- *Corresponding author: X. Luo.*

represent H, a typical LFA model is to minimize a *Loss* function taking the following form [6]: $L(P,Q)=\sum_{h_{m,n}\in H_O} l(\Delta_{m,n})$, $\Delta_{m,n}=h_{m,n}-\hat{h}_{m,n}(p_m,q_n)$, where P and Q represents the two desired latent feature mactrices *w.r.t.* row entities and column entities respectively, H_O denotes the observed entry sets of H, $l(\cdot)$ denotes the function of computing the *loss* based on different L_p-norms, $\hat{h}_{m,n}(\cdot)$ denotes the prediction function of $h_{m,n}$ on different vector spaces, and p_m and q_n denotes the m-th row-vector of P and the n-th row-vector of Q, respectively. Next, we analyze the limitations of an LFA model that adopts a single-metric representation strategy to form its *Loss* function as follows.

- **Different vector spaces comparison (**Top panel of Fig.1**).** Two vector spaces can be adopted to construct $\hat{h}_{m,n}(p_m,q_n)$, i.e., *inner product* and *distance*. Concretely, the inner product has the form of $\hat{h}_{m,n}(p_m,q_n)=p_m q_n$, while that of distance is $\hat{h}_{m,n}(p_m,q_n)=||p_m-q_n||_2$. Supposing the predictions of both $h_{1,2}$ and $h_{2,1}$ are 2.2. Then, the inner product and distance yield different P and Q respectively, which are visualized on the two-dimensional distribution maps. The particular example shows that: 1) the global similarity among m_1, m_2, and m_3 is well represented by inner product space because p_1, p_2, and p_3 are closer to each other, and 2) the local similarity between n_1 and n_2 is better represented by distance space because q_1 and q_2 are closer. Detailed experiments are conducted in Section 4.4.1 to support this finding.
- **Different L_p-norms comparison (**Lower panel of Fig.1**).** Three L_p-norms can be adopted to construct the function $l(\cdot)$, i.e., L_1-norm with $l(\Delta_{m,n})=|\Delta_{m,n}|$, L_2-norm with $l(\Delta_{m,n})=(\Delta_{m,n})^2$, and smooth L_1-norm with $l(\Delta_{m,n}) = \begin{cases} |\Delta_{m,n}|, & if |\Delta_{m,n}|>\omega \\ (\Delta_{m,n})^2, & if |\Delta_{m,n}|\le\omega \end{cases}$, respectively, where ω is a controlling threshold. The bottom of Fig.1 shows their functional relationships and their fitting comparison. The fitting prediction is produced on inner product space with outlier m_4 existing in training. From these figures, we note that: 1) when $|\Delta_{m,n}|>1$, L_1-norm is less sensitive to $\Delta_{m,n}$ than L_2-norm, making L_1-norm-oriented prediction is more robust to outliers than that of L_2-norm; 2) when $\Delta_{m,n}$ closes to zero, L_2-norm is smoother than L_1-norm, i.e., the gradient update of L_2-norm can be much smaller than that of L_1-norm, making L_2-norm-oriented prediction is more stable than that of L_1-norm; 3) the robustness and stability of smooth L_1-norm-oriented prediction are somewhere in between that of L_1-norm and L_2-norm. Detailed experiments are conducted in Section 4.4.2 to support this finding.

Collectively, the above Example 1 substantiates that an HDI matrix usually entails complex relationships between entities, and an LFA model with a single-metric representation strategy cannot reflect them in a comprehensive and unbiased fashion. Motivated by this, we in this work mainly explore one question: *Can an LFA model enjoy the merits rendered from a set of disparate metric spaces all at once, so that it attains superior representation performance?*

Our affirmative answer to this question provides a multi-metric latent feature (MMLF) model for accurate representation of the HDI matrices. Its main idea is two-fold: a) Employing two vector spaces (inner product and distance) and three L_p-norms (L_1-norm, smooth L_1-norm, and L_2-norm) to develop six LFA variants, each of which possesses a unique metric merit, and b) Aggregating the six LFA variants with a self-adaptive weighting strategy, where an overall performance gain is theoretically guaranteed. As such, the proposed MMLF model can enjoy the multiple merits originating from each and every metric space, empowering it to significantly outperform the related state-of-the-art models in representing HDI matrices.

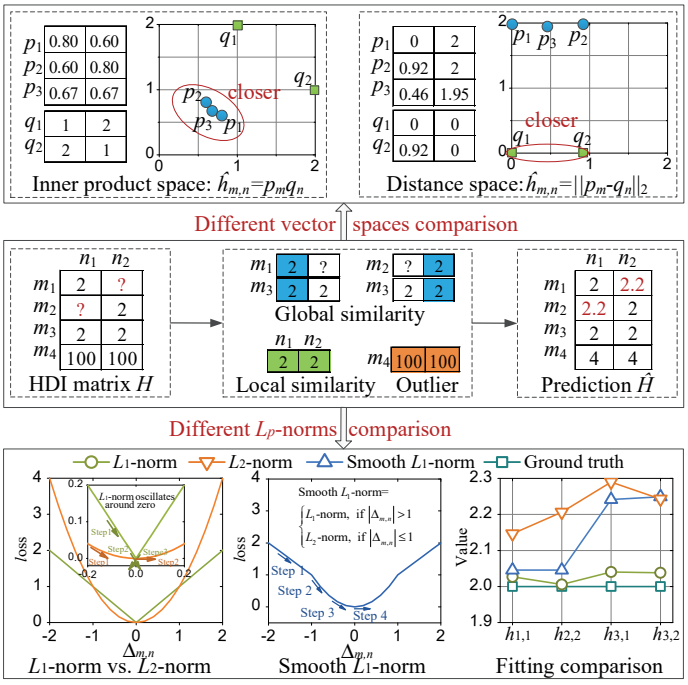


Fig. 1. Example 1: the limitations of an LFA model with a single-metric representation strategy to an HDI matrix.

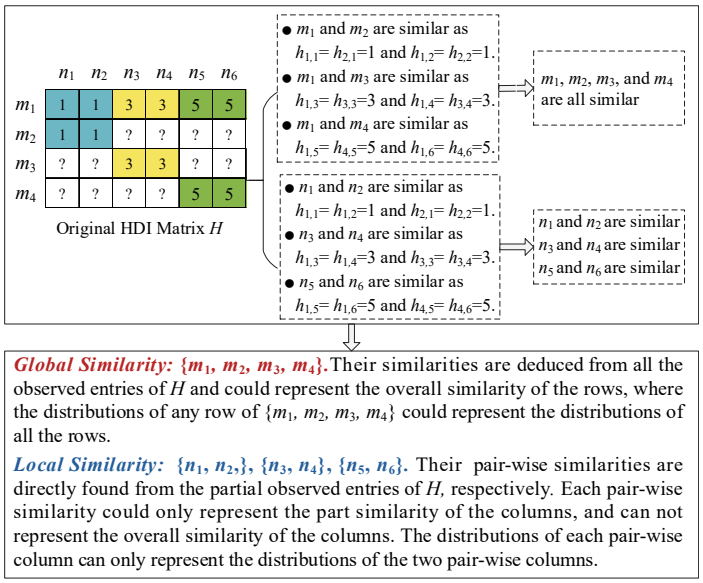


Fig. 2. An example of showing the 'global/local similarity' among row/column entities of an HDI matrix H.

The main contributions of this paper include:

- A novel MMLF model with detailed algorithm design and analysis, which is able to learn accurate representations (i.e., latent features) from an HDI matrix by leveraging a multi-metric latent representation space.
- Multi-metric-based modeling strategy of aggregating six LFA variants which are developed on the two vector spaces and the three L_p-norms, where three variants of them have

also never been proposed in the LFA area.

- Theoretical analyses, which verify that the proposed MMLF can aggregate the merits originating from a set of disparate metric spaces based on the two vector spaces and the three L_p-norms.
- Detailed empirical studies, which explain the reasons why the proposed MMLF can attain performance gain by aggregating the two vector spaces and the three L_p-norms.

Extensive experiments are conducted over ten real-world HDI datasets that span a wide range of applications, including e-commerce, bioinformatics, and fintech. The results suggest that our MMLF outperforms ten competitors, including both deep and shallow counterparts. To promote reproducible research, our source code and Supplementary File are available at the following link ***https://github.com/Wuziqiao/MMLF.git***.

2 Preliminaries and Related Work

2.1 Notations and Symbols

The adopted notations and symbols in this paper are summarized in Table 1.

TABLE 1. Notations.

Notation	Explanation
M,N	Two types of entity sets.
m,n	Single entity of M, N.
$H^{\lvert M\rvert\times\lvert N\rvert}$	Targeted HDI matrix with M rows and N columns.
$h_{m,n}$	An entry of H at m-th row and n-th column.
H_O, H_U	Observed and unobserved entry sets of H.
P, Q	Two latent feature matrices *w.r.t.* M and N, respectively.
p_m, q_n	The m-th row-vector of P and the n-th row-vector of Q, respectively
d	Latent feature dimension.
$\hat{H}$	H's rank-d approximation matrix.
$\hat{h}_{m,n}$	Prediction of $h_{m,n}$.
$\Delta_{m,n}$	Prediction error between $h_{m,n}$ and $\hat{h}_{m,n}$, i.e., $\Delta_{m,n} = h_{m,n}-\hat{h}_{m,n}$.
k	The index of k-th base model of MMLF.
P^k, Q^k	Two latent feature matrices of k-th base model.
p_m^k, q_n^k	m-th row vector of P^k and n-th row vector of Q^k, respectively.
b_m^k, b_n^k	Bias scalars of m and n entities of k-th base model, respectively.
$\hat{h}_{m,n}^k$	Prediction of $h_{m,n}$ by k-th base model.
$\Delta_{m,n}^k$	Prediction error between $h_{m,n}$ and $\hat{h}_{m,n}^k$, i.e., $\Delta_{m,n}^k = h_{m,n}-\hat{h}_{m,n}^k$.
λ,η	Regularization coefficient and learning rate, respectively.
t,T	t-th training iteration, $t\in\{1, \ldots, T\}$, T *is* maximum.
$Pl^k(t)$	The partial *loss*, cumulative *Loss*, and ensemble weight of k-th base model at t-th training iteration, respectively.
$Cl^k(t)$, $\alpha^k(t)$	
$Pl(t)$, $Cl(t)$	The partial *loss*, cumulative *loss* of MMLF at t-th training iteration.
ζ	A balance coefficient controlling the ensemble of base models.
ω	A controlling threshold for smooth L_1-norm.
Γ	Testing set.

2.2 Representation Learning of An HDI Matrix

To describe the representation of an HDI matrix, we first define an HDI matrix.

Definition 1: An HDI matrix. Given two types of entity set M and N, a matrix H with $|M|$ rows and $|N|$ columns records the relationships between M and N, i.e., its each entry $h_{m,n}$ denotes the relationship between entity m ($m\in M$) and entity n ($n\in N$). H_O and H_U denote observed and unobserved entry sets of H, respectively. H is an HDI matrix if M and N are large scales and $|H_O|\ll|H_U|$.

Given an HDI matrix H, its representation learning means embedding its involved row/column entities into the same latent feature space based on H's known data by extracting the low-rank latent features from H, where the extracted latent features contain H's intrinsic patterns and their scale is much smaller than that of H. The extracted latent features could be employed for other downstream data mining tasks, such as quality of service prediction in Web service [16, 17, 54] and rating prediction in recommender systems [6, 7].

2.3 The LFA–based Model

The LFA model is one of the most frequently adopted methods to represent an HDI matrix [6]. Given an HDI matrix H, an LFA model is to learn two latent feature matrices to make H's low-rank approximation $\hat{H}$ by minimizing the differences between H and $\hat{H}$ on H's known entries H_O only. The two matrices are the extracted low-rank latent features from H.

In the past decade, many advancements of LFA-based models have been proposed from different perspectives [1, 8]. In particular, most of them adopt an L_2-norm-oriented Loss, including dual-regularization-based [15], covering-based and neighborhood-aware [17], generalized non-negative and momentum-based [7], neighborhood-and-location integrated [16], joint recommendation [18], content features-based [19], confidence-driven [20], and a generalized non-negative based [21], among others. Nevertheless, L_2-norm is known to be sensitive to outliers [6]. To counter this issue, subsequent researchers propose to adopt L_1-norm to improve an LFA model's robustness to outliers [6, 22, 23]. However, all the above models are developed on inner product vector space that is prone to ignoring fine-grained similarity (i.e., local similarity) [13, 14]. Alternatively, Hsieh et al. propose to apply metric learning [14] to recommender systems, which can capture the locally user-wise or item-wise similarities represented by Euclidean distance directly. On this foundation, Zhang et al. investigate the metric vector space (i.e., distance vector space) to develop an LFA model for recommendation tasks [13].

Notably, all the above LFA-based models are developed with a single-metric representation strategy for HDI data. Recalling Example 1, however, one metric representation strategy commonly has its unique limitations. As a result, they cannot comprehensively represent a target HDI matrix's characteristics. For example, the most adopted strategy is inner product vector space with L_2-norm that is prone to ignore the local similarity and is more sensitive to outlier data. In comparison, our MMLF adopts the multi-metric representation strategy by aggregating the multi-merits of inner product space, distance space, L_1-norm, smooth L_1-norm, and L_2-norm together, which makes it achieve highly accurate representation on an HDI matrix. Besides, three base models developed in our MMLF also have never been reported by prior studies to the authors' best knowledge.

2.4 The Deep Learning-based Model

Recently, deep neural networks (DNNs) [24] have attracted extensive attention in analyzing HDI data from recommender systems due to their powerful representation learning ability [25]. Zhang et al. provide a comprehensive review of DNNs-

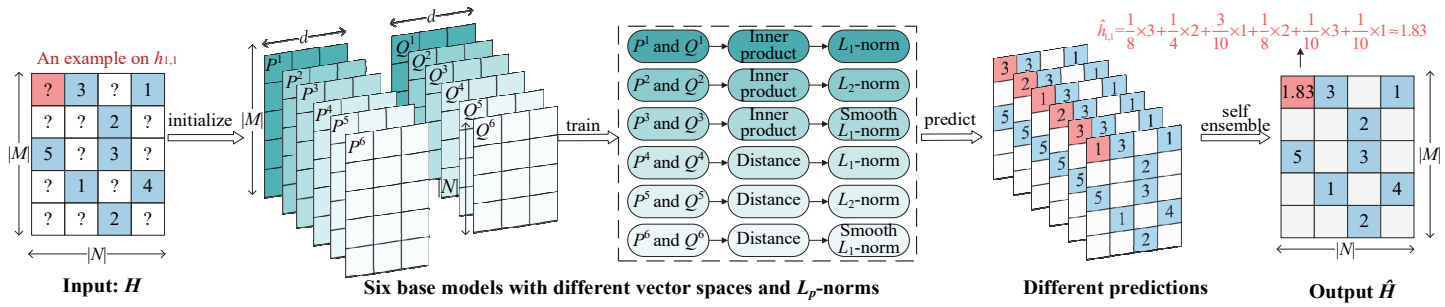


Fig. 3. The overall structure of the proposed MMLF model.

based recommender systems [1]. To date, various sophisticated DNNs-based models have been proposed. Representative studies include autoencoder-based [26], variational autoencoder-based [27], neural collaborative filtering-based [25], neural factorization-based [28], federated meta-learning-based [29], neural rating regression-based [30], deep-rating and review-neural-network-based [31], and latent relevant metric learning-based [32] models. Moreover, with the development of graph neural networks (GNNs) [33-35], some researchers also propose to employ GNNs to analyze HDI data from recommender systems, including graph autoencoder-based [36], inductive graph-based [37], multi-component graph-based [38], neural graph collaborative filtering-based [39, 40], and self-supervised graph-based [41] methods.

Compared with the above deep learning-based models, MMLF possesses its significance in the following aspects: 1) DNNs-based ones usually take complete data (i.e., H_O and H_U) rather than observed data (i.e., H_O) of an HDI matrix as input [42, 43] and GNNs-based ones need to construct a graph in advance [34, 35] that requires huge computation. In comparison, MMLF has much higher computational efficiency because it is trained only on observed entries of an input HDI matrix; 2) DNNs-based methods often need to exploit side information such as text comments [1], to fulfill their data-hungriness and to improve their representation learning capability. Similarly, MMLF is also compatible with this information. It can be easily incorporated into MMLF's training process as a regularization constraint; 3) S. Rendle *et al.* and Xu *et al.* verify that the LFA model is still highly competitive with them [44, 55]. MMLF is developed by improving LFA with a multi-metric representation strategy. For these reasons, we advocate that our MMLF is conceptually comparable with the DNN-based LFA models, while craving much less input information. Numerical experiments and comparisons are presented in Section 4.2 to evidence the performance superiority of our MMLF model.

3 The Proposed MMLF Model

3.1 The Overall Structure

Generally, given an HDI matrix $H^{|M|\times|N|}$, an LFA model is modeled to train two latent feature matrices $P^{|M|\times d}$ and $Q^{|N|\times d}$ to make H's rank-d approximation $\hat{H}$ by minimizing the loss $L(P, Q)$ between H and $\hat{H}$ on H_O only, where $d \ll \min\{|M|, |N|\}$ [6, 7, 12]. The loss $L(P, Q)$ is formulated as follows:

$$L(P,Q)=\sum_{h_{m,n}\in H_O} l\left(\Delta_{m,n}\right),\ \ \Delta_{m,n}=h_{m,n}-\hat{h}_{m,n}\left(p_m,q_n\right), \tag{1}$$

where $l(\cdot)$ denotes the function of computing the loss between $h_{m,n}$ and $\hat{h}_{m,n}$, $\hat{h}_{m,n}$ is the entry of $\hat{H}$ at m-th row and n-th column and is the prediction of $h_{m,n}$, $\hat{h}_{m,n}(\cdot)$ denotes the prediction function of $h_{m,n}$, p_m and q_n denote the m-th row-vector of P, and the n-th row-vector of Q, respectively.

TABLE 2. The *Loss* function of six base LFA models

Base model	$\hat{h}_{m,n}(p_m,q_n)$	$l(\Delta_{m,n})$	Characteristic
MMLF-1	$p_m q_n$	$\lvert\Delta_{m,n}\rvert$	Global similarity &Robustness
MMLF-2	$p_m q_n$	$(\Delta_{m,n})^2$	Global similarity & Stability
MMLF-3	$p_m q_n$	$\lvert\Delta_{m,n}\rvert$, *if* $\lvert\Delta_{m,n}\rvert > \omega$ $(\Delta_{m,n})^2$, *if* $\lvert\Delta_{m,n}\rvert \le \omega$	Global similarity & Relative Robustness/Stability
MMLF-4	$\lvert\lvert p_m - q_n \rvert\rvert_2$	$\lvert\Delta_{m,n}\rvert$	Local similarity & Robustness
MMLF-5	$\lvert\lvert p_m - q_n \rvert\rvert_2$	$(\Delta_{m,n})^2$	Local similarity & Stability
MMLF-6	$\lvert\lvert p_m - q_n \rvert\rvert_2$	$\lvert\Delta_{m,n}\rvert$, *if* $\lvert\Delta_{m,n}\rvert > 1$ $(\Delta_{m,n})^2$, *if* $\lvert\Delta_{m,n}\rvert \le 1$	Local similarity & Relative Robustness/Stability

$\lvert\lvert\cdot\rvert\rvert_2$ denotes the L_2-norm of a vector, ω is a controlling threshold.

Note that Example 1 substantiates that different vector spaces and L_p-norms have different advantages and disadvantages in representing an HDI matrix. Thus, we first pairwise employ the two vector spaces and the three L_p-norms to design the functions of $l(\Delta_{m,n})$ and $\hat{h}_{m,n}(p_m,q_n)$ to build six base LFA models, as shown in Table 2.

To aggregate all the advantages of the six base models, we propose the MMLF model. Fig. 3 depicts its architecture which can be divided into two parts: 1) employing the observed entries of input H (i.e., H_O) to train six base models to predict its missing entries, respectively; 2) self-ensembling the predictions of the six base models by a weighting strategy to output the final $\hat{H}$. To illustrate the principle of MMLF, an example of predicting $h_{1,1}$ is also given in Fig. 3. The predicted values of the six base models are 3, 2, 1, 2, 3, and 1, respectively. They are weighted to obtain the final prediction 1.83.

3.2 Building Base Models

According to [8, 9], linear bias and regularization are crucial for an LFA model to correct bias and avoid overfitting. By incorporating Tikhonov regularization and training bias into (1), we have the following objective function for each base model:

$$\varepsilon(P,Q)=\sum_{h_{m,n}\in H_O} l\left(h_{m,n}-\hat{h}_{m,n}\left(p_m,q_n\right)-b_{m,n}\left(p_m,q_n\right)\right)+\frac{\lambda}{2}\left(\|P\|_F^2+\|Q\|_F^2\right), \tag{2}$$

where λ is a regularization coefficient controlling the penalty intensity, $b_{m,n}(\cdot)$ denotes the bias function *w.r.t.* $h_{m,n}$, $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Generally, (2) is minimized by stochastic gradient descent (SGD) [8, 9] to obtain the training rules of P and Q as follows:

$$for\ h_{m,n}\in H_O:\begin{cases} p_m \leftarrow p_m-\eta\dfrac{\partial\varepsilon\left(P,Q\right)}{\partial p_m} \\ q_n \leftarrow q_n-\eta\dfrac{\partial\varepsilon\left(P,Q\right)}{\partial q_n} \end{cases}, \tag{3}$$

where η is learning rate. Next, we detail six base LFA models.

3.2.1 MMLF-1 (inner product space and L_1-norm)

Following (2) and Table 2, we design the objective function of MMLF-1 with training bias as follows:

$$\varepsilon\left(P^{k=1},Q^{k=1}\right)=\underbrace{\sum_{h_{m,n}\in H_O}\left|h_{m,n}-\underbrace{p_m^{k=1}q_n^{k=1}}_{\text{Inner product space}}-\underbrace{\left(b_m^{k=1}+b_n^{k=1}\right)}_{\text{Training bias}}\right|}_{L_1\text{-norm oriented }loss} +\underbrace{\frac{1}{2}\lambda\sum_{h_{m,n}\in H_O}\left(\left(p_m^{k=1}\right)^2+\left(q_n^{k=1}\right)^2+\left(b_m^{k=1}\right)^2+\left(b_n^{k=1}\right)^2\right)}_{\text{Regularization}}. \tag{4}$$

Following (3), we adopt SGD to minimize (4) to obtain the training rules of $P^{k=1}$ and $Q^{k=1}$ *of* MMLF-1 as follows:

$$\text{for } h_{m,n}\in H_O:\begin{cases} \Delta_{m,n}^{k=1}\geq 0:\begin{cases} p_m^{k=1}\leftarrow\left(1-\eta\lambda\right)p_m^{k=1}+\eta q_n^{k=1} \\ q_n^{k=1}\leftarrow\left(1-\eta\lambda\right)q_n^{k=1}+\eta p_m^{k=1} \\ b_m^{k=1}\leftarrow\left(1-\eta\lambda\right)b_m^{k=1}+\eta \\ b_n^{k=1}\leftarrow\left(1-\eta\lambda\right)b_n^{k=1}+\eta \end{cases} \\ \Delta_{m,n}^{k=1}<0:\begin{cases} p_m^{k=1}\leftarrow\left(1-\eta\lambda\right)p_m^{k=1}-\eta q_n^{k=1} \\ q_n^{k=1}\leftarrow\left(1-\eta\lambda\right)q_n^{k=1}-\eta p_m^{k=1} \\ b_m^{k=1}\leftarrow\left(1-\eta\lambda\right)b_m^{k=1}-\eta \\ b_n^{k=1}\leftarrow\left(1-\eta\lambda\right)b_n^{k=1}-\eta \end{cases} \end{cases}, \tag{5}$$

where $\Delta_{m,n}^{k=1}=h_{m,n}-p_m^{k=1}q_n^{k=1}-b_m^{k=1}-b_n^{k=1}$.

3.2.2 MMLF-2 (inner product space and L_2-norm)

Following (2) and Table 2, we design the objective function of MMLF-2 with training bias as follows:

$$\varepsilon\left(P^{k=2},Q^{k=2}\right)=\underbrace{\frac{1}{2}\sum_{h_{m,n}\in H_O}\left(h_{m,n}-p_m^{k=2}q_n^{k=2}-b_m^{k=2}-b_n^{k=2}\right)^2}_{L_2\text{-norm oriented }loss} +\frac{1}{2}\lambda\sum_{h_{m,n}\in H_O}\left(\left(p_m^{k=2}\right)^2+\left(q_n^{k=2}\right)^2+\left(b_m^{k=2}\right)^2+\left(b_n^{k=2}\right)^2\right). \tag{6}$$

Following (3), we adopt SGD to minimize (6) to obtain the training rules of $P^{k=2}$ and $Q^{k=2}$ *of* MMLF-2 as follows:

$$\text{for } h_{m,n}\in H_O:\begin{cases} p_m^{k=2}\leftarrow\left(1-\eta\lambda\right)p_m^{k=2}+\eta\Delta_m^{k=2}q_n^{k=2} \\ q_n^{k=2}\leftarrow\left(1-\eta\lambda\right)q_n^{k=2}+\eta\Delta_{m,n}^{k=2}p_m^{k=2} \\ b_m^{k=2}\leftarrow\left(1-\eta\lambda\right)b_m^{k=2}+\eta\Delta_{m,n}^{k=2} \\ b_n^{k=2}\leftarrow\left(1-\eta\lambda\right)b_n^{k=2}+\eta\Delta_{m,n}^{k=2} \end{cases}, \tag{7}$$

where $\Delta_{m,n}^{k=2}=h_{m,n}-p_m^{k=2}q_n^{k=2}-b_m^{k=2}-b_n^{k=2}$.

3.2.3 MMLF-3 (inner product space and smooth L_1-norm)

Following (2) and Table 2, we design the objective function of MMLF-3 with training bias as follows:

$$\varepsilon\left(P^{k=3},Q^{k=3}\right)=\underbrace{\sum_{h_{m,n}\in H_O}\begin{cases}\left|\Delta_{m,n}^{k=3}\right|, & if\ \left|\Delta_{m,n}^{k=3}\right|>\omega \\ \left(\Delta_{m,n}^{k=3}\right)^2, & if\ \left|\Delta_{m,n}^{k=3}\right|\leq\omega\end{cases}}_{\text{Smooth }L_1\text{-norm oriented }loss} +\frac{1}{2}\lambda\sum_{h_{m,n}\in H_O}\left(\left(p_{m,}^{k=3}\right)^2+\left(q_n^{k=3}\right)^2+\left(b_m^{k=3}\right)^2+\left(b_n^{k=3}\right)^2\right), \tag{8}$$

where $\Delta_{m,n}^{k=3}=h_{m,n}-p_m^{k=3}q_n^{k=3}-b_m^{k=3}-b_n^{k=3}$. Following (3), we adopt SGD to minimize (8) to obtain the training rules of $P^{k=3}$ and $Q^{k=3}$ *of* MMLF-3 as follows:

$$\text{for } h_{m,n}\in H_O:\begin{cases} \Delta_{m,n}^{k=3}>\omega:\begin{cases} p_m^{k=3}\leftarrow\left(1-\eta\lambda\right)p_m^{k=3}+\eta q_n^{k=3} \\ q_n^{k=3}\leftarrow\left(1-\eta\lambda\right)q_n^{k=3}+\eta p_m^{k=3} \\ b_m^{k=3}\leftarrow\left(1-\eta\lambda\right)b_m^{k=3}+\eta \\ b_n^{k=1}\leftarrow\left(1-\eta\lambda\right)b_n^{k=3}+\eta \end{cases} \\ \Delta_{m,n}^{k=3}<-\omega:\begin{cases} p_m^{k=3}\leftarrow\left(1-\eta\lambda\right)p_m^{k=3}-\eta q_n^{k=3} \\ q_n^{k=3}\leftarrow\left(1-\eta\lambda\right)q_n^{k=3}-\eta p_m^{k=3} \\ b_m^{k=3}\leftarrow\left(1-\eta\lambda\right)b_m^{k=3}-\eta \\ b_n^{k=3}\leftarrow\left(1-\eta\lambda\right)b_n^{k=3}-\eta \end{cases} \\ \left|\Delta_{m,n}^{k=3}\right|\leq\omega:\begin{cases} p_m^{k=3}\leftarrow\left(1-\eta\lambda\right)p_m^{k=3}+2\eta\Delta_{m,n}^{k=3}q_n^{k=3} \\ q_n^{k=3}\leftarrow\left(1-\eta\lambda\right)q_n^{k=3}+2\eta\Delta_{m,n}^{k=3}p_m^{k=3} \\ b_m^{k=3}\leftarrow\left(1-\eta\lambda\right)b_m^{k=3}+2\eta\Delta_{m,n}^{k=3} \\ b_n^{k=3}\leftarrow\left(1-\eta\lambda\right)b_n^{k=3}+2\eta\Delta_{m,n}^{k=3} \end{cases} \end{cases}. \tag{9}$$

3.2.4 MMLF-4 (distance space and L_1-norm)

Following (2) and Table 2, we design the objective function of MMLF-4 with training bias as follows:

$$\varepsilon\left(P^{k=4},Q^{k=4}\right)=\sum_{h_{m,n}\in H_O}\left|h_{m,n}-\underbrace{\left\|p_m^{k=4}-q_n^{k=4}\right\|_2}_{\text{Distance space}}-b_m^{k=4}-b_n^{k=4}\right| +\frac{1}{2}\lambda\sum_{h_{m,n}\in H_O}\left(\left(p_m^{k=4}\right)^2+\left(q_n^{k=4}\right)^2+\left(b_m^{k=4}\right)^2+\left(b_n^{k=4}\right)^2\right). \tag{10}$$

Following (3), we adopt SGD to minimize (10) to obtain the training rules of $P^{k=4}$ and $Q^{k=4}$ *of* MMLF-4 as follows:

$$\text{for } h_{m,n}\in H_O:\begin{cases} \Delta_{m,n}^{k=4}\geq 0:\begin{cases} p_m^{k=4}\leftarrow\left(1-\eta\lambda\right)p_m^{k=4}+\eta\dfrac{p_m^{k=4}-q_n^{k=4}}{\left\|p_m^{k=4}-q_n^{k=4}\right\|_2} \\ q_n^{k=4}\leftarrow\left(1-\eta\lambda\right)q_n^{k=4}-\eta\dfrac{p_m^{k=4}-q_n^{k=4}}{\left\|p_m^{k=4}-q_n^{k=4}\right\|_2} \\ b_m^{k=4}\leftarrow\left(1-\eta\lambda\right)b_m^{k=4}+\eta \\ b_n^{k=4}\leftarrow\left(1-\eta\lambda\right)b_n^{k=4}+\eta \end{cases} \\ \Delta_{m,n}^{k=4}<0:\begin{cases} p_m^{k=4}\leftarrow\left(1-\eta\lambda\right)p_m^{k=4}-\eta\dfrac{p_m^{k=4}-q_n^{k=4}}{\left\|p_m^{k=4}-q_n^{k=4}\right\|_2} \\ q_n^{k=4}\leftarrow\left(1-\eta\lambda\right)q_n^{k=4}+\eta\dfrac{p_m^{k=4}-q_n^{k=4}}{\left\|p_m^{k=4}-q_n^{k=4}\right\|_2} \\ b_m^{k=4}\leftarrow\left(1-\eta\lambda\right)b_m^{k=4}-\eta \\ b_n^{k=4}\leftarrow\left(1-\eta\lambda\right)b_n^{k=4}-\eta \end{cases} \end{cases}, \tag{11}$$

where $\Delta_{m,n}^{k=4}=h_{m,n}-||p_m^{k=4}-q_n^{k=4}||_2-b_m^{k=4}-b_n^{k=4}$.

3.2.5 MMLF-5 (distance space and L_2-norm)

Following (2) and Table 2, we design the objective function of MMLF-5 with training bias as follows:

$$\varepsilon\left(P^{k=4},Q^{k=4}\right)=\frac{1}{2}\sum_{h_{m,n}\in H_O}\left(h_{m,n}-\left\|p_m^{k=4}-q_n^{k=4}\right\|_2-b_m^{k=4}-b_n^{k=4}\right)^2 +\frac{1}{2}\lambda\sum_{h_{m,n}\in H_O}\left(\left(p_m^{k=4}\right)^2+\left(q_n^{k=4}\right)^2+\left(b_m^{k=4}\right)^2+\left(b_n^{k=4}\right)^2\right). \tag{12}$$

Following (3), we adopt SGD to minimize (12) to obtain the

training rules of $P^{k=5}$ and $Q^{k=5}$ *of* MMLF-5 as follows:

$$\text{for } h_{m,n}\in H_O:\begin{cases} p_m^{k=5}\leftarrow(1-\eta\lambda)p_m^{k=5}+\eta\Delta_m^{k=5}\dfrac{p_m^{k=5}-q_n^{k=5}}{\left\|p_m^{k=5}-q_n^{k=5}\right\|_2}\\ q_n^{k=5}\leftarrow(1-\eta\lambda)q_n^{k=5}-\eta\Delta_{m,n}^{k=5}\dfrac{p_m^{k=5}-q_n^{k=5}}{\left\|p_m^{k=5}-q_n^{k=5}\right\|_2}\\ b_m^{k=5}\leftarrow(1-\eta\lambda)b_m^{k=5}+\eta\Delta_{m,n}^{k=5}\\ b_n^{k=5}\leftarrow(1-\eta\lambda)b_n^{k=5}+\eta\Delta_{m,n}^{k=5}\end{cases}, \quad (13)$$

where $\Delta_{m,n}^{k=5}=h_{m,n}-\|p_m^{k=5}-q_n^{k=5}\|_2-b_m^{k=5}-b_n^{k=5}$.

3.2.6 MMLF-6 (distance space and smooth L_1-norm)

Following (2) and Table 2, we design the objective function of MMLF-6 with training bias as follows:

$$\begin{aligned}\varepsilon\left(P^{k=6},Q^{k=6}\right)=&\sum_{h_{m,n}\in H_O}\begin{pmatrix}\left|\Delta_{m,n}^{k=6}\right|, & if\ \left|\Delta_{m,n}^{k=6}\right|>\omega\\ \left(\Delta_{m,n}^{k=6}\right)^2, & if\ \left|\Delta_{m,n}^{k=6}\right|\le\omega\end{pmatrix}\\ &+\frac{1}{2}\lambda\sum_{h_{m,n}\in H_O}\left(\left(p_{m,}^{k=6}\right)^2+\left(q_n^{k=6}\right)^2+\left(b_m^{k=6}\right)^2+\left(b_n^{k=6}\right)^2\right),\end{aligned} \quad (14)$$

where $\Delta_{m,n}^{k=6}=h_{m,n}-\|p_m^{k=6}-q_n^{k=6}\|_2-b_m^{k6}-b_n^{k=6}$. Following (3), we adopt SGD to minimize (14) to obtain the training rules of $P^{k=6}$ and $Q^{k=6}$ *of* MMLF-6 as follows:

$$\text{for } h_{m,n}\in H_O:$$

$$\begin{cases}\Delta_{m,n}^{k=6}>\omega:\begin{cases} p_m^{k=6}\leftarrow(1-\eta\lambda)p_m^{k=6}+\eta\dfrac{p_m^{k=6}-q_n^{k=6}}{\left\|p_m^{k=6}-q_n^{k=6}\right\|_2}\\ q_n^{k=6}\leftarrow(1-\eta\lambda)q_n^{k=6}-\eta\dfrac{p_m^{k=6}-q_n^{k=6}}{\left\|p_m^{k=6}-q_n^{k=6}\right\|_2}\\ b_m^{k=6}\leftarrow(1-\eta\lambda)b_m^{k=6}+\eta\\ b_n^{k=6}\leftarrow(1-\eta\lambda)b_n^{k=6}+\eta\end{cases}\\ \Delta_{m,n}^{k=6}<-\omega:\begin{cases} p_m^{k=6}\leftarrow(1-\eta\lambda)p_m^{k=6}-\eta\dfrac{p_m^{k=6}-q_n^{k=6}}{\left\|p_m^{k=6}-q_n^{k=6}\right\|_2}\\ q_n^{k=6}\leftarrow(1-\eta\lambda)q_n^{k=6}+\eta\dfrac{p_m^{k=6}-q_n^{k=6}}{\left\|p_m^{k=6}-q_n^{k=6}\right\|_2}\\ b_m^{k=6}\leftarrow(1-\eta\lambda)b_m^{k=6}-\eta\\ b_n^{k=6}\leftarrow(1-\eta\lambda)b_n^{k=6}-\eta\end{cases}\\ \left|\Delta_{m,n}^{k=6}\right|\le\omega:\begin{cases} p_m^{k=6}\leftarrow(1-\eta\lambda)p_m^{k=6}+2\eta\Delta_{m,n}^{k=6}\dfrac{p_m^{k=6}-q_n^{k=6}}{\left\|p_m^{k=6}-q_n^{k=6}\right\|_2}\\ q_n^{k=6}\leftarrow(1-\eta\lambda)q_n^{k=6}-2\eta\Delta_{m,n}^{k=6}\dfrac{p_m^{k=6}-q_n^{k=6}}{\left\|p_m^{k=6}-q_n^{k=6}\right\|_2}\\ b_m^{k=6}\leftarrow(1-\eta\lambda)b_m^{k=6}+2\eta\Delta_{m,n}^{k=6}\\ b_n^{k=6}\leftarrow(1-\eta\lambda)b_n^{k=6}+2\eta\Delta_{m,n}^{k=6}\end{cases}\end{cases}. \quad (15)$$

3.3 Self-Aggregation

Ensemble learning is a great way to aggregate multi-models. It requires the base models to be diversified and accurate [45, 46]. In MMLF, the six base models are built by the different two vector spaces and three L_p-norms, guaranteeing their diversity. Moreover, since an LFA model is demonstrated to be accurate [6, 7] and these base models are the variant of LFA, which guarantees the accuracy of the base models in representing the HDI matrix (Detailed experiments are conducted in Section 4.3.2 to support this point). Hence, they satisfy the two requirements of ensemble learning. To finely ensemble them, we make their weights self-adaptive according to the prediction *loss* on the validation set. The main idea is to increase *k*-th base model's weight if its partial *loss* decreases at *t*-th training iteration, and decrease its weight otherwise. To theoretically validate the effectiveness of this strategy, we first present the following definitions.

***Definition* 2.** Let $Pl^k(t)$ be the partial *loss* of *k*-th base model at *t*-th training iteration, then it can be computed as follows:

$$\begin{aligned}&Pl^k(t)=\sum_{h_{m,n}\in H_O}\left|h_{m,n}-\hat{h}_{m,n}^k\right|,\\ &\hat{h}_{m,n}^k=p_m^kq_n^k+b_m^k+b_n^k,\qquad s.t.\ k=1,2,3,\\ &\hat{h}_{m,n}^k=\left\|p_m^k-q_n^k\right\|_2+b_m^k+b_n^k,\ s.t.\ k=4,5,6.\end{aligned} \quad (16)$$

***Definition* 3.** Let $Cl^k(t)$ be the cumulative *loss* of $Pl^k(t)$ until *t*-th training iteration, then it can be computed as follows:

$$Cl^k(t)=\sum_{i=1}^{t}Pl^k(t). \quad (17)$$

***Definition* 4.** Let $\alpha^k(t)$ be the ensemble weight of *k*-th base model at *t*-th training iteration, then it can be set as follows:

$$\alpha^k(t)=\frac{e^{-\zeta Cl^k(t)}}{\sum_{k=1}^{6}e^{-\zeta Cl^k(t)}}, \quad (18)$$

where ζ is a balance coefficient controlling the ensemble of base models during the training processes.

Then, based on definitions 2–4, the final predictions of MMLF at *t*-th training iteration are obtained as follows:

$$\hat{h}_{m,n}=\sum_{k=1}^{6}\alpha^k(t)\hat{h}_{m,n}^k \quad (19)$$

3.4 Theoretical Analysis

This section theoretically analyzes how the proposed MMLF can aggregate the merits originating from a set of disparate metric spaces based on the two vector spaces and the three L_p-norms. First, we introduce the related definitions, theorem, proposition, and remark. Then, the corresponding proofs are provided.

***Definition* 5.** Let $Pl(t)$ be the *loss* of MMLF model at *t*-th training iteration, then it can be computed as follows:

$$Pl(t)=\sum_{h_{m,n}\in H_O}\left|h_{m,n}-\hat{h}_{m,n}\right|, \quad (20)$$

where $\hat{h}_{m,n}$ is obtained by (19).

***Definition* 6.** Let $Cl(t)$ be the cumulative *loss* of $Pl(t)$ until *t*-th training iteration, then it can be computed as follows:

$$Cl(t)=\sum_{i=1}^{t}Pl(t) \quad (21)$$

***Theorem* 1**. Considering the MMLF model, assuming that its base modes' $Pl^k(t)$ lies in the scale of [0,1]. If $\alpha^k(t)$ is set as (18) during the training processes, then the following equality holds:

$$Cl(T)\le\min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}+\frac{\ln 6}{\zeta}+\frac{\zeta T}{8}, \quad (22)$$

where T is the maximum number of iterations.

In *Theorem* 1, by setting $\zeta=\sqrt{1/\ln T}$, the upper bound becomes $\min\{Cl^k(T)\,|\,k=1,2,3,4,5,6\}+\ln 6\sqrt{\ln T}+T/(8\sqrt{\ln T})$, where the term of

$\ln 6\sqrt{\ln T}+T/(8\sqrt{\ln T})$ is linearly bounded by the number of iterations. Then, we have the following proposition.

***Proposition* 1.** By setting $\zeta=\sqrt{1/\ln T}$, the following inequality holds:

$$Cl(T)\le \min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}+const, \tag{23}$$

where $\lim_{T\to\infty} const=19.45$.

***Remark* 1**. *Proposition* 1 states that $Cl(T)$ is bounded by $\min\{Cl^k(T)\,|\,k=1,2,3,4,5,6\}+const$ with the condition of $\zeta=\sqrt{1/\ln T}$ and $T\to\infty$. In other words, *Proposition* 1 guarantees that by setting the ensemble weight as (18), MMLF's cumulative training *loss* is always comparable to, or not larger than, that of each base model during the training process. Notably, each of six base models adopts a unique representation metric, which makes each of them have a unique merit from the characteristics of inner product space, distance space, L_1-norm, smooth L_1-norm, and L_2-norm. Therefore, *Proposition* 1 shows that MMLF possesses multi-merits originating from the two vector spaces and the three L_p-norms (please refer to Section 4.4.1 to find the different merits of the two vector spaces, and refer to Section 4.4.2 to find the different merits of the three L_p-norms).

3.4.1 Proof of Theorem 1

To begin with, let us recall the *Hoeffding Inequality* [47]:

***Lemma* 1**. Let X be a random variable fulfilling $a\le X\le b$, $\forall s\in\mathbb{R}$, the following inequality holds:

$$\ln \mathrm{E}\left[e^{sX}\right]\le s\mathrm{E}X+\frac{s^2(b-a)^2}{8}. \tag{24}$$

The detailed proof of *Lemma* 1 is given in [48].□

Then, $Cl^k(t)$ are bounded by:

$$A_t=\sum_{k=1}^{6} e^{-\zeta Cl^k(t)} \tag{25}$$

and $A_0=6$ since there is no *loss* when $t=0$. Based on (25), we have the following inference:

$$\ln\frac{A_T}{A_0}\ge -\zeta\min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}-\ln 6. \tag{26}$$

Besides, $\forall t\in\{1, 2, \ldots, T\}$, based on (18) and (25), we have:

$$\ln\frac{A_t}{A_{t-1}}=\ln\left(\sum_{k=1}^{6}\alpha^k(t)e^{-\zeta Cl^k(t)}\right). \tag{27}$$

Let $s=-\zeta$ and $X\in\{Pl^k(t)\,|\,k=1,2,3,4,5,6,\ s.t.\ Pl^k(t)\in[0,1]\}$. $\alpha^k(t)$ is interpreted as the probability of $Pl^k(t)$. Thus, we have:

$$s\mathrm{E}X=-\zeta\sum_{k=1}^{6}\alpha^k(t)Pl^k(t). \tag{28}$$

Then based on *Lemma* 1, (19), (20), (27), and (28), we have

$$\ln\frac{A_t}{A_{t-1}}\le -\zeta\sum_{k=1}^{6}\alpha^k(t)Pl^k(t)+\frac{\gamma^2}{8}=-\zeta Pl(t)+\frac{\gamma^2}{8}. \tag{29}$$

Considering the accumulation of (29) as t increases from 1 to T, we have the following inferences:

$$\ln\frac{A_T}{A_0}\le -\zeta Cl(T)+\frac{\zeta^2}{8}T. \tag{30}$$

By combining (26) and (30), the following inequality is achieved:

$$Cl(T)\le \min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}+\frac{\ln 6}{\zeta}+\frac{\zeta T}{8}. \tag{31}$$

Then, *Theorem* 1 holds.□

3.4.2 Proof of Proposition 1

To begin with, let us recall the *Bernstein Inequality* [48]:

***Lemma* 2**. Let X be a random variable in the scale of [0,1], then $\forall s\in\mathbb{R}$, the following inequality holds:

$$\ln \mathrm{E}\left[e^{sX}\right]\le\left(e^s-1\right)\mathrm{E}X. \tag{32}$$

The detailed proof of *Lemma* 2 is given in [48].□

By combining *Lemma* 2, (19), (20), (27), and (28), the following inequality is achieved:

$$\ln\frac{A_t}{A_{t-1}}\le\left(e^{-\xi}-1\right)Pl(t). \tag{33}$$

Considering the accumulation of (33) according to (31) as t increases from 1 to T, we have the following inferences:

$$\ln\frac{A_T}{A_0}\le\left(e^{-\zeta}-1\right)Cl(T). \tag{34}$$

By combining (26) and (34), we achieve:

$$Cl(T)\le\frac{\zeta}{1-e^{-\zeta}}\min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}+\frac{\ln 6}{1-e^{-\zeta}}. \tag{35}$$

Let $\psi=\zeta/(1-e^{-\zeta})$ and $\Phi=\ln 6/(1-e^{-\zeta})$, then we rewrite (35) as $Cl(T)\le\psi\min\{Cl^k(T)\,|\,k=1,2,3,4,5,6\}+\Phi$. With $\zeta=\sqrt{1/\ln T}$, we have $\lim_{T\to\infty}\psi=1$ following L'Hopital's Rule [49]. Moreover, considering the partial derivative of Φ with T, we can infer that $\lim_{T\to\infty}(\mathrm{d}\Phi/\mathrm{d}T)\to 0$. Hence, Φ becomes a constant (i.e., 19.45 according to its curve) when $T\to\infty$. Finally, (35) is reduced to:

$$\begin{aligned} Cl(T)&\le\left[\min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}\right]^{+}+\left[19.45\right]^{-}\\ &\le\min\left\{Cl^k(T)\middle|k=1,2,3,4,5,6\right\}+\underset{<19.45}{const}. \end{aligned} \tag{36}$$

Then, we see that (36) holds with $\zeta=\sqrt{1/\ln T}$ and $T\to\infty$. Therefore, *Proposition* 1 holds.□

3.5 Algorithm Design

TABLE 3. ALGORITHM MMLF

Steps	**Input:** H_O, **Output:** $\hat{H}$	**Cost**
1	initialize d, λ, η, ζ, $\alpha^k(0)=1/6$, $T=2000$	$\Theta(1)$
2	initialize P^k of each base model randomly;	$\Theta(\|M\|\times d\times 6)$
3	initialize Q^k of each base model randomly;	$\Theta(\|N\|\times d\times 6)$
4	initialize b_m^k of each base model randomly, $m\in\{1,\ldots,\|M\|\}$	$\Theta(\|M\|\times 6)$
5	initialize b_n^k of each base model randomly, $n\in\{1,\ldots,\|N\|\}$	$\Theta(\|N\|\times 6)$
6	**while** $t\le T$ && not converge	$\times T$
7	**for** k=1 to 6	$\times 6$
8	**for** $\forall h_{m,n}\in H_O$	$\times\|H_O\|$
9	update p_m^k and q_n^k according to (5)/(7)/(9)/(11)/(13)/(15)	$\times 2d$
10	**end for**	-
11	**end for**	-
12	**for** k=1 to 6	$\times 6$
13	compute $Cl^k(t)$ according to (16)/(17)	$\times\|H_O\|\times d$
14	**end for**	-
15	compute $\alpha^k(t)$ according to (18)	$\Theta(1)$
16	**for** $\forall h_{m,n}\in H_O$	$\times\|H_O\|$
17	compute prediction $\hat{h}_{m,n}$ according to (19)	$\Theta(1)$
18	**end for**	-
19	$t=t+1$	$\Theta(1)$
20	**end while**	-

According to the above analyses, we design the *Algorithm MMLF*. Table 3 shows its pseudo-code, where the cost of each step is analyzed. Notably, as $(|M|+|N|)\ll|H_O|$ in real-world applications, we deduce that Algorithm MMLF's time complexity is $\Theta(T\times|H_O|\times d)$, which is the same as that of an

LFA model [6, 9]. Besides, considering space complexity, Algorithm MMLF adopts the following data structures: 1) two arrays with length $|H_O|$ to cache observed entries H_O and corresponding predictions, 2) six matrices with size $|M|\times d$ to cache P^k, 3) six matrices with size $|N|\times d$ to cache Q^k, 4) six arrays with length $|M|$ to cache b_m^k for each entity of M, and 5) six arrays with length $|N|$ to cache b_n^k for each entity of N. Then, Algorithm MMLF's space complexity is $O(d\times\max\{|H_O|/d, |M|, |N|\})$. Therefore, MMLF's time and space complexities are efficient because they are both linear with $|H_O|$.

4 Experiments

In the subsequent experiments, we aim at answering the following research questions (RQs):

- RQ. 1. Does the proposed MMLF model outperform state-of-the-art models in representing an HDI matrix?
- RQ. 2. How does MMLF self-adaptively control the ensemble of its base models and what are their influences?
- RQ. 3. What are the reasons that MMLF can attain evident performance gain by aggregating its base models?
- RQ. 4. How does the latent feature dimension d influence the representation learning ability of MMLF?

4.1 General Settings

Datasets. Ten frequently used HDI datasets are selected to conduct the experiments. They are real datasets generated from different fields, including e-commerce, bioinformatics, and fintech. Table 4 summarizes their details. MovieLens_20M, MovieLens_10M, MovieLens_1M, and EachMovie are collected from the movie recommendation website MovieLens[1]. Dating is collected from an online dating website LibimSeTi [6]. Flixter is collected from the Flixter website [50]. 1_zyr_162425 and 1_zyr_224308 are two protein-protein association networks for supporting functional discovery [51]. Bitcoin alpha is to record the trust degree of users in bitcoin transactions [52]. Douban is collected by Douban.com [6].

Evaluation Metrics. Missing data prediction is an important and crucial issue in evaluating the representation of an HDI matrix [6, 9]. To evaluate the accuracy of missing data prediction, we adopt the widely used root mean squared error (RMSE) and mean absolute error (MAE) as the evaluation metrics [1, 6-9]. They are calculated by:

$$RMSE=\sqrt{\left(\sum_{h_{m,n}\in\Gamma}\left(h_{m,n}-\hat{h}_{m,n}\right)^2\right)\Big/|\Gamma|},\ MAE=\left(\sum_{h_{m,n}\in\Gamma}\left|h_{m,n}-\hat{h}_{m,n}\right|\right)\Big/|\Gamma|$$

where Γ denotes the testing set.

Baselines. We compare the proposed MMLF model with nine related state-of-the-art models, including five LFA-based models (BLF, SL-LF, L³F, GFNLF, and FML) and four deep learning-based models (AutoRec, NRR, MetaMF, and NeuMF). Table 5 gives brief descriptions of these competitors.

Implementation Details. We set d=20 for all the LFA-based modes to draw a fair comparison. We respectively tune the learning rate and regularization coefficient for all the models to achieve their own best prediction accuracy. The training of a model terminates if its training iterations reach a preset threshold (i.e., 2000) or the error difference between two consecutive iterations is smaller than 10-6. We adopt 80%–20% train-test settings, where hyper-parameters are pre-tuned on a validation set that is partially split from training data. We test each dataset five times and report the average results. All the experiments are run on a server that has 2.2 GHz E5-2630 CPU with 40 cores, 256 GB RAM, and GTX 1080 Ti GPU. The source code of the proposed MMLF model is available at the following link ***https://github.com/Wuziqiao/MMLF.git***.

TABLE 4. PROPERTIES OF ALL THE DATASETS.

No.	Name	$\|M\|$	$\|N\|$	$\|H_O\|$	Density*
D1	MovieLens_20M	138,493	26,744	20,000,263	0.54%
D2	MovieLens_10M	71,567	65,133	10,000,054	0.21%
D3	Dating	135,359	168,791	17,359,346	0.08%
D4	EachMovie	72,916	1,628	2,811,718	2.37%
D5	Flixter	147,612	48,794	8,196,077	0.11%
D6	1_zyr_162425	7,963	7,963	1,120,027	1.77%
D7	1_zyr_224308	4,181	4,181	1,021,783	5.85%
D8	Bitcoin Alpha	7,604	7,604	24,186	0.04%
D9	Douban	129,490	58,541	16,830,839	0.22%
D10	MovieLens_1M	6,040	3,706	1,000,209	4.47%

*Density denotes the percentage of observed entries in the HDI matrix.

TABLE 5. DESCRIPTIONS OF ALL THE INVOLVED MODELS.

Model	Description
BLF [9]	It is the classic LFA model and has been widely adopted to represent HDI data. It won the notable competition Netflix Prize in 2009.
SL-LF [23]	It is an improved LFA model by adopting a smooth L_1-norm-oriented *loss* to represent HDI data. *IEEE JAS 2021.*
L³F [6]	It is an improved LFA model by adopting an L_1- and L_2-norn oriented *loss* to represent HDI data. *IEEE TNNLS 2021.*
GFNLF [21]	It is a non-negative LFA model by adopting the α-β-divergence to enhance its representation learning ability. *WWW 2020.*
FML [13]	It is an LFA-based model that combines metric learning (distance space) and collaborative filtering. *IEEE TII 2020.*
AutoRec [26]	It is the representative DNNs-based model in representing HDI data from the recommender system. *WWW 2015.*
NRR [30]	It is a DNNs-based multi-task learning framework for rating prediction in a recommender system. *SIGIR 2017.*
MetaMF [29]	Is is a DNNs-based federated learning framework for rating prediction in a recommender system. *SIGIR 2020.*
NeuMF [25]	It is the representative DNNs-based model for item ranking in a recommender system. We replace its objective function with an Euclidean distance-based one for rating prediction. *WWW 2017.*

4.2 Performance Comparison (RQ. 1)

4.2.1 Comparison of Prediction Accuracy

Table 6 records the prediction accuracy of all the involved models on different datasets. To better analyze these results, we make statistical analyses of the loss/win, the Wilcoxon signed-ranks test [53], and the Friedman test [53]. The loss/win is to count how many cases that MMLF has higher/lower RMSE/MAE than each comparison model on all the datasets, respectively. The Wilcoxon signed-ranks test is a nonparametric pairwise comparison method. It checks whether MMLF has significantly higher prediction accuracy than each comparison model by the level of significance p-value. The Friedman test is to compare the performance of multiple models on multiple datasets simultaneously by the F-rank value. A lower F-rank

[1] https://grouplens.org/datasets/movielens/

value denotes a higher prediction accuracy. The statistical re-

TABLE 6. THE COMPARISON RESULTS ON PREDICTION ACCURACY, INCLUDING LOSS/WIN COUNTS, WILCOXON SIGNED-RANKS TEST, AND FRIEDMAN TEST.

Dataset	Metric	BLF	SL-LF	L³F	GFNLF	FML	AutoRec	NRR	MetaMF	NeuMF	MMLF
D1	RMSE	0.7737	0.7790	0.7806	0.7792	0.7802	0.7802	0.7798	0.7905	0.7931	**0.7693**
	MAE	0.5886	0.5857	0.5863	0.5901	0.5905	0.5947	0.6018	0.6221	0.6049	**0.5789**
D2	RMSE	0.7819	0.7887	0.7899	0.7872	0.7892	0.7865	0.7834	0.8010	0.7969	**0.7787**
	MAE	0.5999	0.5980	0.5981	0.6032	0.6037	0.6048	0.6035	0.6169	0.6118	**0.5909**
D3	RMSE	1.8066	1.7829	1.7704	1.8124	1.8012	1.8027	1.8101	1.8013	1.8257	**1.7591**
	MAE	1.2392	1.1686	1.1781	1.2442	1.2412	1.2610	1.2303	1.2406	1.2635	**1.1440**
D4	RMSE	0.2251	0.2260	0.2256	0.2272	0.2258	0.2305	0.2301	0.2365	0.2361	**0.2212**
	MAE	0.1732	0.1729	0.1698	0.1774	0.1713	0.1784	0.1774	0.1813	0.1882	**0.1683**
D5	RMSE	0.8961	0.8324●	0.8326●	0.8790	0.8420	0.8682	0.8677	0.8781	0.8641	0.8382
	MAE	0.6447	0.6084●	0.6050●	0.6394	0.6219	0.6295	0.6337	0.6443	0.6429	0.6089
D6	RMSE	0.1243	0.1232	0.1259	0.1241	0.1242	0.1172●	0.1202	0.1285	0.1405	0.1200
	MAE	0.0856	0.0846	0.0813	0.0848	0.0878	0.0789●	0.0903	0.0953	0.1026	0.0803
D7	RMSE	0.1248	0.1250	0.1289	0.1275	0.1258	0.1198●	0.1333	0.1368	0.1432	0.1222
	MAE	0.0839	0.0837	0.0802	0.0845	0.0876	0.0801	0.0920	0.0955	0.1046	**0.0795**
D8	RMSE	0.2638	0.2641	0.2609	0.2812	0.2651	0.2807	0.2642	0.2843	0.3027	**0.2529**
	MAE	0.1595	0.1585	0.1511	0.1619	0.1607	0.1625	0.1517	0.1455	0.1854	**0.1453**
D9	RMSE	0.7192	0.6957	0.7000	0.7142	0.7177	0.7080	0.7106	0.7201	0.7183	**0.6949**
	MAE	0.5563	0.5458	0.5347	0.5574	0.5611	0.5606	0.5675	0.5728	0.5688	**0.5341**
D10	RMSE	0.8497	0.8491	0.8459	0.8471	0.8493	0.8475	0.8711	0.8794	0.8694	**0.8445**
	MAE	0.6692	0.6601	0.6593	0.6621	0.6672	0.6671	0.6713	0.6892	0.6782	**0.6564**
	loss/win	0/20	2/18	2/18	0/20	0/20	3/17	0/20	0/20	0/20	**7/173***
Statistic	*p*-value	**4.78×10⁻⁵**	**2.77×10⁻⁴**	**5.44×10⁻⁴**	**4.78×10⁻⁵**	**4.78×10⁻⁵**	**1.82×10⁻⁴**	**4.78×10⁻⁵**	**4.78×10⁻⁵**	**4.78×10⁻⁵**	-
	F-rank	5.4	3.4	3.5	6.025	5.825	5.425	6.275	8.65	9.15	**1.35**

● The cases that MMLF loses the comparison. * The total loss/win cases of MMLF.

sults of the loss/win, the Wilcoxon signed-ranks test, and the Friedman test are presented at the third-to-last, second-to-last, and last rows of Table 6, respectively. From Table 6, we have the following important observations:

- MMLF achieves the highest estimation accuracy for missing data of an HDI matrix on 15 testing cases out of 20 in total on all the ten datasets. Moreover, its total loss/win cases situation is 7/173 when comparing each model one by one.
- All *p*-values are much smaller than 0.05, which denotes that MMLF has significantly higher prediction accuracy than all the comparison models with a significance level of 0.05.
- MMLF achieves the lowest F-rank value among all the models, further verifying that it has the highest prediction accuracy on the tested datasets.

Therefore, these observations demonstrate that MMLF significantly outperforms the comparison models in terms of accuracy in predicting the missing data of an HDI matrix.

4.2.2 Comparison of Computational Efficiency

Fig. 4 records all the involved models' CPU running time of one training, where we have two main observations:

- The DNNs-based models consume much more time than the LFA-based models. The reason is that DNNs-based ones are trained on the complete data of an HDI matrix. In comparison, LFA-based ones are trained only on the observed data of an HDI matrix.
- MMLF has no advantage compared with other LFA-based models. The reason is that MMLF needs to train six base models. However, they can be parallel trained as they are independent of each other in each training iteration. For example, by parallel training the six base models of MMLF, its CPU running time is decreased from 6266 to 1135 *secs*, which is comparable to that of LFA-based models (i.e., BLF is 615, SL-LF is 715, L³F is 1232, GFNLF is 2352, and FML is 187587).

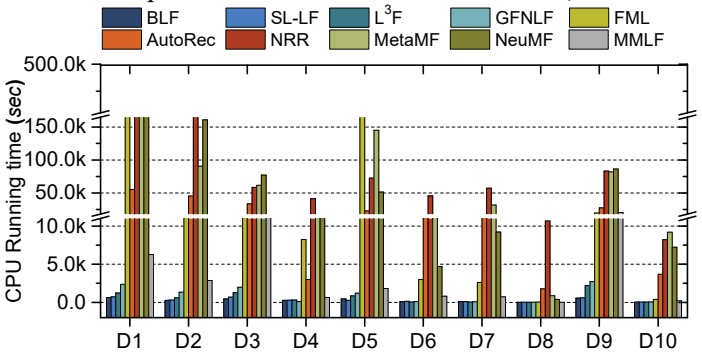


Fig. 4. The comparison of the running time of all the models.

Therefore, these two observations demonstrate that MMLF's computational efficiency is much higher than those of DNNs-based models and comparable to those of most efficient LFA-based models.

4.3 Self-adaptive Ensemble and Ablation Experiments (RQ. 2)

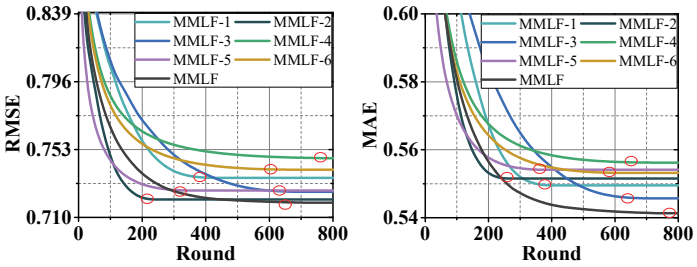
4.3.1 Self-adaptive Ensemble

First, we test the convergence of MMLF and its each base model. The results on D1 are presented in Fig. 5. The complete results on all the datasets are recorded in Figs. S1–S2 in the Supplementary File. We see that both MMLF and its each base model are convergent and they have different convergent rounds. For example, when testing RMSE on D1, the convergent rounds of MMLF-1, MMLF-2, MMLF-3, MMLF-4, MMLF-5, MMLF-6, and MMLF are about 383, 213,

624, 778, 312, 601, and 638, respectively. Note that since MMLF's base models are independent with each other, we adjust their training round on the validation data that is partially split from training data. By ensembling the six base models, MMLF can converge to a lower RMSE/MAE on most cases.

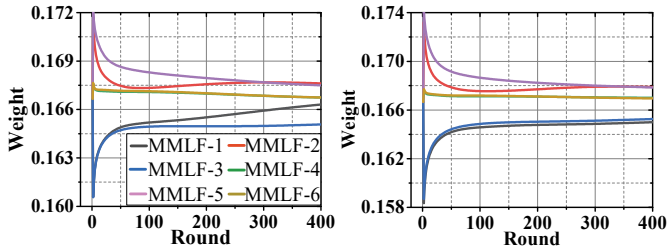
Table 7. THE COMPARISON RESULTS OF MMLF BETWEEN FIXED WEIGHT AND SELF-ADAPTIVE WEIGHT.

Dataset	D1		D2		D3		D4		D5		D6		D7		D8		D9		D10	
Metric	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
Fixed weight	0.7712	0.5808	0.7823	0.5951	1.7799	1.1565	0.2218	0.1687	0.8422	0.6128	0.1302	0.0878	0.1327	0.0861	0.2569	0.1484	0.6991	0.5380	0.8450	0.6568
Adaptive weight	0.7693	0.5789	0.7787	0.5909	1.7591	1.144	0.2212	0.1683	0.8382	0.6089	0.1200	0.0803	0.1222	0.0795	0.2529	0.1453	0.6949	0.5341	0.8445	0.6564
Improvement (%)	**0.25%**	**0.33%**	**0.46%**	**0.71%**	**1.17%**	**1.08%**	**0.27%**	**0.24%**	**0.47%**	**0.64%**	**7.83%**	**8.54%**	**7.91%**	**7.67%**	**1.56%**	**2.09%**	**0.60%**	**0.72%**	**0.06%**	**0.06%**



(a)RMSE (b)MAE

Fig. 5. The converging curves of MMLF and its each base model on D1, where the red circle denotes the convergent point.



(a)D1 (b)D2

Fig. 6. The changes of ensemble weights during the training process.

Second, to empirically study how MMLF self-adaptively controls the ensemble of its base models, we monitor the changes of ensemble weights during the training process. In the beginning, six base models are initialized with the same weights. The results on D1 and D2 are presented in Fig. 6. The complete results on all the datasets are recorded in Fig. S3 in the Supplementary File. From these results, we find that all the weights adaptively change during the training process. Finally, they converge to the different constants. For example, on D1, the weights of MMLF-1 and MMLF-3 dramatically decrease first and then gradually increase until convergence. In terms of MMLF-2, MMLF-4, MMLF-5, and MMLF-6, the changes of weights are opposite. Finally, all the weights converge to 0.1663, 0.1676, 0.1651, 0.1667, 0.1675, and 0.1668, respectively.

Finally, to verify the positive influence of self-adaptive controls of ensemble weights, we compare MMLF between fixed weight (i.e., equal weight for each base model) and self-adaptive weight. The results are presented in Table 7, where we find that MMLF with adaptive weight consistently achieves better prediction accuracy than that with fixed weight. The accuracy improvement ranges from 0.06% to 8.54%, which verifies that MMLF's self-adaptive weight strategy can guarantee its performance.

Summary. This set of experiments validates that MMLF can self-adaptively control the ensembles of its base models to converge during the training process. Consequently, such a self-adaptive ensemble makes it possess the multi-merits of its base models to guarantee its highly accurate representations to an HDI matrix.

4.3.2 Ablation Experiments

This sub-section studies how MMLF's base models influence its performance through ablation experiments. To this end, we conduct two sub-sets of experiments as follows.

The first is to compare MMLF with its each base model. The results are recorded in Table 8. Similarly, we also conduct the loss/win counts, the Wilcoxon signed-ranks test [53], and the Friedman test [53] on these comparison results. From Table 8, we have three observations: 1) Although the base models have relatively worse accuracy than MMLF model, they are still accurate in representing HDI data, 2) The base models have different performances on the different datasets, no one can guarantee to perform best on all the datasets. For example, MMLF-1 has lower RMSE/MAE than MMLF-4 on D1, while the results are opposite on D5; 2) MMLF achieves the lowest RMSE/MAE on all the datasets except for only three cases. It significantly outperforms its six base models.

The second is to compare MMLF with its degraded versions that exclude one base model from itself. The comparison results are recorded in Table 9, where we have two discoveries: 1) Each base model has different influences on MMLF. For example, excluding MMLF-1 and MMLF-4 makes MMLF perform worse on MAE while excluding MMLF-2 and MMLF-5 makes MMLF perform worse on RMSE, and excluding MMLF-3 and MMLF-6 makes MMLF perform differently on the different datasets. 2) MMLF significantly outperforms the degraded versions of excluding MMLF-2 and MMLF-5, and slightly outperforms the other degraded versions.

Summary. From the two sub-sets of experiments, we can conclude that: 1) since MMLF's each base model has different characteristics, they perform differently on the different datasets and no one can guarantee to perform best on all the datasets, 2) MMLF's each base model is crucial for boosting MMLF to perform better on all the datasets.

4.4 Performance Gain Analysis (RQ. 3)

This section aims to analyze the reasons why MMLF can attain evident performance gain by ensembling its base models. To this end, we conduct the experiments from three aspects: 1)

comparing the influences between inner product space and distance space, 2) comparing the influences between L_1-norm and L_2-norm, and 3) comparing the distributions of MMLF's base models.

TABLE 8. THE COMPARISON RESULTS BETWEEN MMLF AND ITS ONLY ONE BASE MODEL, RESPECTIVELY.

Dataset	Metric	Including only one base model						MMLF
		MMLF-1	MMLF-2	MMLF-3	MMLF-4	MMLF-5	MMLF-6	
D1	RMSE	0.7852	0.7714	0.7764	0.7979	0.7768	0.7910	0.7693
	MAE	0.5875	0.5892	0.5849	0.5938	0.5916	0.5900	0.5789
D2	RMSE	0.7923	0.7801	0.7846	0.8074	0.7863	0.8000	0.7787
	MAE	0.5968	0.6001	0.5948	0.6073	0.6032	0.6030	0.5909
D3	RMSE	1.8501	1.8043	1.8439	1.8380	1.7790	1.8355	1.7591
	MAE	1.1445	1.2488	1.1636	1.1676	1.2469	1.1841	1.1440
D4	RMSE	0.2311	0.2223	0.2224	0.2322	0.2236	0.2236	0.2212
	MAE	0.1729	0.1705	0.1706	0.1737	0.1719	0.1718	0.1683
D5	RMSE	0.8699	0.8539	0.8630	0.8603	0.8385	0.8536	0.8382
	MAE	0.6211	0.6354	0.6244	0.6129	0.6224	0.6150	0.6089
D6	RMSE	0.1316	0.1237	0.1235	0.1355	0.1290	0.1292	0.1200
	MAE	0.0813	0.0867	0.0866	0.0860	0.0926	0.0926	0.0803
D7	RMSE	0.1352	0.1252	0.1253	0.1372	0.1303	0.1300	0.1222
	MAE	0.0803	0.0863	0.0864	0.0839	0.0928	0.0925	0.0795
D8	RMSE	0.2692	0.2624	0.2681	0.2687	0.2533	0.2548	0.2529
	MAE	0.1440●	0.1617	0.1530	0.1425●	0.1542	0.1536	0.1453
D9	RMSE	0.7322	0.7032	0.7073	0.7039	0.7309	0.7115	0.6949
	MAE	0.5290●	0.5526	0.5505	0.5536	0.5420	0.5546	0.5341
D10	RMSE	0.8578	0.8457	0.8491	0.8717	0.8509	0.8637	0.8445
	MAE	0.6611	0.6642	0.6601	0.6717	0.6685	0.6726	0.6564
	loss/win	2/18	0/20	0/20	1/19	0/20	0/20	**3/117**
Statistic	*p*-value	**2.41×10^{-4}**	**4.78×10^{-5}**	**4.78×10^{-5}**	**5.57×10^{-5}**	**4.78×10^{-5}**	**4.78×10^{-5}**	-
	F-rank	4.45	3.75	3.65	5.35	4.65	5	**1.15**

TABLE 9. THE COMPARISON RESULTS BETWEEN MMLF AND ITS BASE MODELS WHEREIN ONLY ONE IS EXCLUDED, RESPECTIVELY.

Dataset	Metric	Excluding only one base model						MMLF
		MMLF-1	MMLF-2	MMLF-3	MMLF-4	MMLF-5	MMLF-6	
D1	RMSE	0.7692●	0.7726	0.7697	0.7678●	0.7710	0.7685●	0.7693
	MAE	0.5797	0.5795	0.5794	0.5792	0.5789	0.5793	0.5789
D2	RMSE	0.7784●	0.7823	0.7795	0.7769●	0.7802	0.7778●	0.7787
	MAE	0.5915	0.5917	0.5916	0.5909	0.5908●	0.5910	0.5909
D3	RMSE	1.7582●	1.7766	1.7588●	1.7628	1.7822	1.7635	1.7591
	MAE	1.1540	1.1390●	1.1507	1.1540	1.1423●	1.1524	1.1440
D4	RMSE	0.2214	0.2224	0.2224	0.2212	0.2220	0.2220	0.2212
	MAE	0.1692	0.1689	0.1690	0.1690	0.1687	0.1687	0.1683
D5	RMSE	0.8356●	0.8398	0.8362●	0.8392	0.8422	0.8399	0.8382
	MAE	0.6091	0.6068●	0.6080●	0.6121	0.6093	0.6112	0.6089
D6	RMSE	0.1204	0.1206	0.1207	0.1199●	0.1203	0.1202	0.1200
	MAE	0.0823	0.0801●	0.0801●	0.0817	0.0797●	0.0797●	0.0803
D7	RMSE	0.1221●	0.1228	0.1227	0.1216●	0.1228	0.1228	0.1222
	MAE	0.0813	0.0792●	0.0791●	0.0809	0.0789●	0.0790●	0.0795
D8	RMSE	0.2521●	0.2559	0.2517●	0.2522●	0.2540	0.2538	0.2529
	MAE	0.1472	0.1446●	0.1457	0.1472	0.1447●	0.1447●	0.1453
D9	RMSE	0.6940●	0.7006	0.6992	0.6943●	0.7007	0.6996	0.6949
	MAE	0.5414	0.5340●	0.5328●	0.5408	0.5337●	0.5322●	0.5341
D10	RMSE	0.8446	0.8485	0.8456	0.8422●	0.8456	0.8447	0.8445
	MAE	0.6593	0.6596	0.6589	0.6571	0.6575	0.6559●	0.6564
	loss/win	7/13	6/14	7/13	7/13	6/14	7/13	**40/80**
Statistic	*p*-value	0.051	**0.012**	0.078	0.069	**0.019**	0.112	-
	F-rank	4.35	5.1	4.15	3.575	4.15	3.6	**3.075**

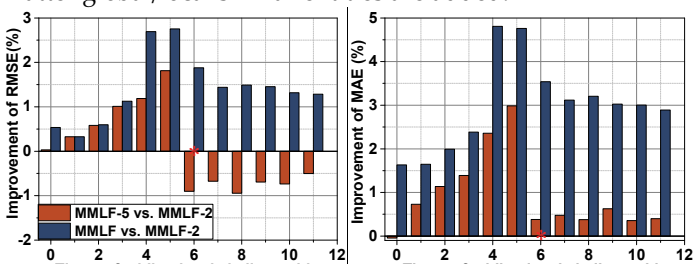
4.4.1 Influence Comparison Between Inner Product and Distance Spaces

We generate synthetic datasets based on a real HDI dataset of *MovieLens Latest* (*https://grouplens.org/datasets/movielens/)* that has 610 rows and 9742 columns to simulate the different situations of global/local similarity. The specific generating method is: a) randomly selecting 100 rows out from 610 rows as the basic dataset, b) generating a column corresponding to each original column with the same known values, c) randomly changing 5% known maximum values of each generated synthetic column, d) generating the times of local similar columns from 1 to 11. To illustrate this method, an example is given in Fig. S4 in the Supplementary File.

Fig. 7 records the results of the performance comparison between inner product and distance spaces *w.r.t.* L_2-norm. The other comparison results *w.r.t.* L_1-norm and smooth L_1-norm are recorded in Fig. S5 in the Supplementary File. Note that MMLF-1, MMLF-2, and MMLF-3 represent inner product space, MMLF-4, MMLF-5, and MMLF-6 represent distance space, and MMLF represents both spaces. From these results, we have the following findings::

- In the beginning, the inner product models and distance space models perform similarly. For example, MMLF-5 has a 0.03% lower RMSE and a 0.05% higher MAE than MMLF-2.
- As the added local similar entities increase, distance space models perform better and better. However, when the added entities reach a threshold (marked as red ∗), distance space models can not receive evident performance gain, or even are deteriorated. For example, the results between MMLF-5 vs. MMLF-2 show that the improvement of RMSE is increased from 0.03% to 1.81% when the times of adding entities are increased from 1 to 5. After that, the improvement of RMSE is decreased into the range [-0.9%, -0.5%].
- The comparisons between MMLF and inner product models show that MMLF can receive evident performance gain no matter how many local similar entities are added. However, the improvement has some degradation after the added entities reach a threshold (marked as red ∗).

These observations show that distance space models can receive evident performance gain as local similar entities increase. However, after the added entities reach a threshold, such gain does not exist anymore. The reason is that when numerous local similar entities are added, they turn to be the global similar entities. Therefore, this set of experiments substantiates that distance space tends to represent local similarity while inner product space tends to represent global similarity on HDI data. Notably, since our MMLF possesses the merits of both inner product and distance spaces, it can receive evident performance gain on all the tested cases no matter global/local similar entities are added.



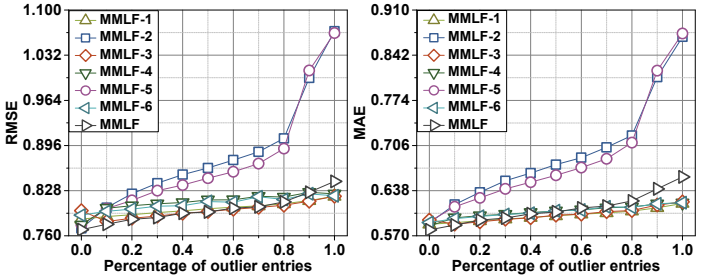
(a)RMSE (b)MAE

Fig. 7. The performance comparison between inner product and distance spaces *w.r.t.* L_2-norm, where the red $*$ denotes the inflection point of performance improvement.

4.4.2 Influence Comparison between L_1-norm and L_2-norm

We generate synthetic outlier data on all the adopted datasets to test the robustness of MMLF and its each base model. The concrete method of generating outlier data is as follows: a) an unknown entry of the input HDI matrix between two known entries is randomly selected as the outlier entry, b) assigning a maximum or minimum value to the outlier entity, c) increasing the percentage that outlier entries account for known entries from 0% to 100% with a 10% interval, and d) only adding the outlier entries into the training set. An example is provided to illustrate this method in Fig. S6 in the Supplementary File.

Fig. 8 records the results on D1. The complete results on all the datasets are recorded in Fig. S7–S8 in the Supplementary File. We observe that at the beginning, MMLF-2 and MMLF-5 perform slightly better than MMLF-1, MMLF-3, MMLF-4, and MMLF-6. However, as the percentage of outlier data increases, MMLF-2 and MMLF-5 perform worse and worse than MMLF-1, MMLF-3, MMLF-4, and MMLF-6. For example on D1, the RMSE of (MMLF-2 and MMLF-5) and (MMLF-1, MMLF-3, MMLF-4, and MMLF-6) is (0.7714, 0.7768) and (0.7852, 0.7979, 0.7764, and 0.791), respectively, when there are no outlier data; and then is (1.0681 and 1.0654) and (0.8195, 0.8194, 0.8232, and 0.8203), respectively, when the percentage of outlier data is 100%. Therefore, these results substantiate that L_1-norm-based and smooth L_1-norm-based LFA models are much more robust to outlier data than L_2-norm—based ones. Notably, since our MMLF possesses all the merits of L_1-norm, L_2-norm, and smooth L_1-norm, it performs well in all the tested cases no matter how much outlier data are added.



(a)RMSE (b)MAE

Fig. 8. The influence of outlier data to MMLF and its base models on D1.

4.4.3 The Distribution Comparison of Latent Features of MMLF's Base Models

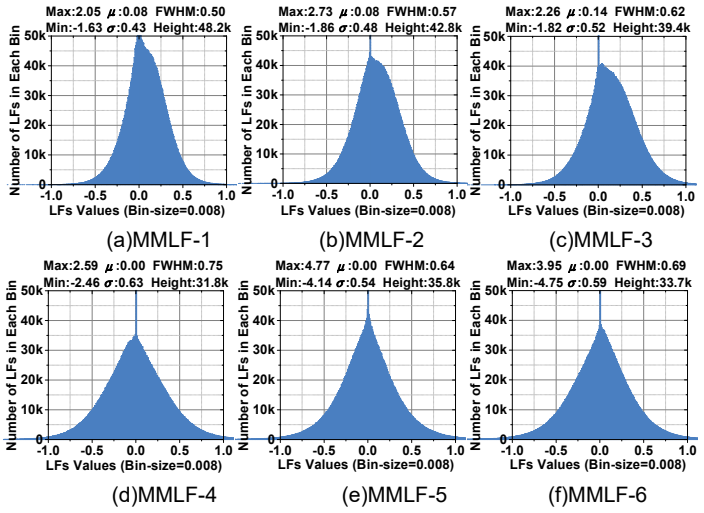
To check whether MMLF's base models are diversified in representing an identical HDI matrix, we analyze their distributions of latent features. Fig. 9 depicts the distributions of latent features on D1. The distributions of latent features on other datasets are depicted in Fig. S9–S17 in the Supplementary File. To analyze these figures, we adopt a Gaussian function

$$f(v) = \theta_1 + \frac{\theta_2}{\sigma\sqrt{\pi/2}} e^{-2\frac{(v-\mu)^2}{\sigma^2}}$$

to fit them, where v is the value of the latent feature, $f(v)$ is the number of latent features in each bin, θ_1 and θ_2 are the constants, μ is the expectation, and σ is the standard deviation. The full width at half maximum (FWHM) and height of the Gaussian curve are measured. Besides, we also record the maximum/minimum of latent features. All the statistical parameters are inserted in these figures. Then, we have the following observations:

- The distributions of inner product space tend to be more centralized than that of distance space. For example on D1, the ranges between maximum and minimum inner product space are 3.68, 4.59, and 4.08, respectively, which are smaller than that of distance space of 5.05, 8.91, and 8.70, respectively. Besides, the FWHM of inner product space are 0.50, 0.57, and 0.62, respectively, while that of distance space are 0.75, 0.64, and 0.69, respectively.
- The distributions of distance space tend to be symmetric *w.r.t.* zero while that of inner product space does not. For example on D1, the μ of distance space is zero, while that of inner product space are 0.08, 0.08, and 0.14, respectively.
- Among inner product or distance space, the amplitudes of distributions are different. For example on D1, the heights of MMLF-1, MMLF-2, and MMLF-3 are 48.2k, 42.8k, and 39.4k on inner product space.

These observations obviously reveal that the distributions of latent features of base models are significantly different, demonstrating that the base models are diversified.



(a)MMLF-1 (b)MMLF-2 (c)MMLF-3

(d)MMLF-4 (e)MMLF-5 (f)MMLF-6

Fig. 9. The distribution histogram of latent features (LFs) of MMLF's each base model on D1.

4.4.4 Reasons for MMLF's Performance Gain

From the experimental results of this section, we can conclude three reasons that make MMLF attain evident performance gain: 1) MMLF can represent both global and local similarity well by ensembling inner product and distance spaces, 2) MMLF is robust to outlier data by ensembling L_1-norm and smooth L_1-norm, and 3) MMLF's base models meet the two requirements of ensemble learning, i.e., they are diversified and accurate.

4.5 Influence of Latent Feature Dimension d (RQ. 4)

This set of experiments evaluates the prediction accuracy of MMLF as d varies from 5 to 80. The results on D1–D2 are

presented in Fig. 10. The complete results on all the datasets are recorded in Fig. S18 in the Supplementary File. We can see that the larger d makes MMLF achieve a lower RMSE/MAE, which indicates that the larger d can improve MMLF's representation learning ability. The reason is that the larger d makes MMLF have more latent features to represent an HDI matrix [6, 9, 23]. However, when d is larger than 40, the improvement tends to be slight. According to Section 3.4, MMLF's time complexity linearly increases *w.r.t.* d. Hence, d should be appropriately set to carefully balance representation learning ability and computational cost.

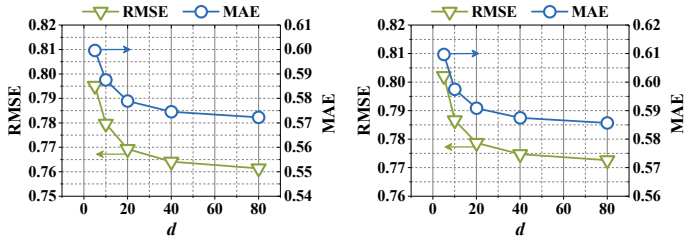


Fig. 10 The RMSE/MAE of MMLF as d increases from 5 to 80 on D1–D2.

5 Conclusion

This paper proposes a multi-metric latent feature (MMLF) model for highly accurate representation of high-dimensional and incomplete (HDI) matrices. Its main idea is two-fold: 1) employing two vector spaces and three L_p-norms to develop six variants of latent feature analysis (LFA) model, where each one has a different metric representation strategy, and 2) aggregating these variants with a carefully designed adaptive weight strategy. Theoretical and empirical studies show that MMLF can self-adaptively aggregate the multi-merits originated from the two vector spaces and the three L_p-norms. Experimental results on ten real datasets collected from a variety of scientific and industrial domains demonstrate that 1) MMLF has significantly higher accuracy in predicting the missing data of an HDI matrix, 2) its computational efficiency is much higher than those of deep learning-based models and comparable to those of most efficient LFA-based models, and 3) it can attain performance gain from a set of disparate metric spaces all at once by aggregating the two vector spaces and the three L_p-norms.

In the future, we plan to conduct theoretical studies regarding the issue of MMLF's ability to establish progressive unbiased estimations for missing data of an HDI matrix. Besides, we plan to apply MMLF to real data mining areas, such as web service, social networks, recommender systems, and end-edge-cloud computing [56].

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China under grants 62176070, 62272078, and 62106024.

References

[1] Z. Zheng, L. Xiaoli, M. Tang, F. Xie, and M. R. Lyu, "Web service QoS prediction via collaborative filtering: a survey," *IEEE Transactions on Services Computing*, vol. 15, no. 4, pp. 2455-2472, 2022.

[2] D. Wu, P. Zhang, Y. He, and X. Luo, "A Double-Space and Double-Norm Ensembled Latent Factor Model for Highly Accurate Web Service QoS Prediction," *IEEE Transactions on Services Computing*, vol. 16, no. 2, pp. 802-814, 2023.

[3] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: a survey and new perspectives," *Acm Computing Surveys*, vol. 52, no. 1, Feb, 2019.

[4] M. Toprak, C. Boldrini, A. Passarella and M. Conti, "Harnessing the Power of Ego Network Layers for Link Prediction in Online Social Networks," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 1, pp. 48-60, 2023.

[5] X. Wang, M. Chen, V. C. M. Leung, Z. Han and K. Hwang, "Integrating Social Networks with Mobile Device-to-Device Services," *IEEE Transactions on Services Computing*, vol. 14, no. 4, pp. 1209-1223, 2021.

[6] D. Wu, M. Shang, X. Luo, and Z. Wang, "An L_1-and-L_2-norm-oriented latent factor model for recommender systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 10, pp.5775-5788, 2022.

[7] X. Luo, Y. Zhou, Z. Liu and M. Zhou, "Fast and accurate non-negative latent factor analysis on high-dimensional and sparse matrices in recommender systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3897-3911, 2023.

[8] P. Zhang, F. Xiong, H. Leung, and W. Song, "FunkR-pDAE: Personalized Project Recommendation Using Deep Learning," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 2, pp. 886-900, 2021.

[9] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, no. 8, pp. 30-37, 2009.

[10] A. Flanagan, W. Oyomno, A. Grigorievskiy, K. E. Tan, S. A. Khan, and M. Ammad-ud-din, "Federated Multi-view Matrix Factorization for Personalized Recommendations," *In Proceedings of the European Conference, ECML PKDD 2020*, 2020, pp. 324-347.

[11] X. He, J. Tang, X. Du, R. Hong, T. Ren, and T.-S. Chua, "Fast Matrix Factorization With Nonuniform Weights on Missing Data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 8, pp. 2791-2804, 2020.

[12] W. Cheng, Y. Shen, Y. Zhu, and L. Huang, "DELF: A Dual-Embedding based Deep Latent Factor Model for Recommendation," *In Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI*, 2018, pp. 3329-3335.

[13] S. Zhang, L. Yao, B. Wu, X. Xu, X. Zhang, and L. Zhu, "Unraveling Metric Vector Spaces With Factorization for Recommendation," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 732-742, 2020.

[14] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," *In Proceedings of the 26th International Conference on World Wide Web, ACM WWW*, 2017, pp. 193-201.

[15] H. Wu, Z. Zhang, K. Yue, B. Zhang, J. He, and L. Sun, "Dual-regularized matrix factorization with deep neural networks for recommender systems," *Knowledge-Based Systems*, vol. 145, pp. 46-58, 2018.

[16] D. Ryu, K. Lee, and J. Baik, "Location-Based Web Service QoS Prediction via Preference Propagation to Address Cold Start Problem," *IEEE Transactions on Services Computing*, vol. 14, no. 3, pp. 736-746, 2021.

[17] Y. Zhang, K. Wang, Q. He, F. Chen, S. Deng, Z. Zheng, and Y. Yang, "Covering-Based Web Service Quality Prediction via Neighborhood-Aware Matrix Factorization," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1333-1344, 2021.

[18] H. Liu, L. Jing, J. Yu, and M. K. Ng, "Social recommendation with learning personal and social latent factors," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 7, pp. 2956-2970, 2021.

[19] H. Zhang, Y. Sun, M. Zhao, T. W. S. Chow, and Q. M. J. Wu, "Bridging User Interest to Item Content for Recommender Systems: An Optimization Model," *IEEE Transactions on Cybernetics*, vol. 50, no. 10,

pp. 4268-4280, 2020.

[20] C. Wang, Q. Liu, R. Wu, E. Chen, C. Liu, X. Huang, and Z. Huang, "Confidence-aware matrix factorization for recommender systems," *In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 434-442.

[21] Y. Yuan, X. Luo, M. Shang, and D. Wu, "A Generalized and Fast-converging Non-negative Latent Factor Model for Predicting User Preferences in Recommender Systems," *In Proceedings of The Web Conference 2020, ACM WWW, Taipei, Taiwan*, 2020, pp. 498–507.

[22] X. Zhu, X. Y. Jing, D. Wu, Z. He, J. Cao, D. Yue, and L. Wang, "Similarity-Maintaining Privacy Preservation and Location-Aware Low-Rank Matrix Factorization for QoS Prediction Based Web Service Recommendation," *IEEE Transactions on Services Computing,* vol. 14, no. 3, pp. 889-902, 2021.

[23] D. Wu, and X. Luo, "Robust Latent Factor Analysis for Precise Representation of High-Dimensional and Sparse Data," *IEEE/CAA Journal of Automatica Sinica,* vol. 8, no. 4, pp. 796-805, 2021.

[24] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[25] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," *In Proceedings of the 26th International World Wide Web Conference*, *ACM WWW,* 2017, pp. 173-182.

[26] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec:Autoencoders Meet Collaborative Filtering," *In Proceedings of the 24th International Conference on World Wide Web*, *ACM WWW*, 2015, pp. 111-112.

[27] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational Autoencoders for Collaborative Filtering," *In Proceedings of the 2018 World Wide Web Conference on World Wide Web, ACM WWW, Lyon, France,* 2018, pp. 689-698.

[28] X. He, and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," *In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017, pp. 355-364.

[29] Y. Lin, P. Ren, Z. Chen, Z. Ren, D. Yu, J. Ma, M. d. Rijke, and X. Cheng, "Meta Matrix Factorization for Federated Rating Predictions," *In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 981-990.

[30] P. Li, Z. Wang, Z. Ren, L. Bing, and W. Lam, "Neural rating regression with abstractive tips generation for recommendation," *In Proceedings of the 40th International conference on Research and Development in Information Retrieval, ACM SIGIR* 2017, pp. 345-354.

[31] W. D. Xi, L. Huang, C. D. Wang, Y. Y. Zheng, and J. H. Lai, "Deep Rating and Review Neural Network for Item Recommendation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 11, pp. 6726-6736, 2022.

[32] Y. Tay, L. Anh Tuan, and S. C. Hui, "Latent relational metric learning via memory-based attention for collaborative ranking," *In Proceedings of the 2018 World Wide Web Conference*, *ACM WWW,* 2018, pp. 729-739.

[33] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 32, no. 1, pp. 4-24, 2021.

[34] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?," *In Proceedings of the 7th International Conference on Learning Representations, ICLR*, 2019.

[35] W. Fan, Y. Ma, Q. Li, Y. He, Y. E. Zhao, J. Tang, and D. Yin, "Graph Neural Networks for Social Recommendation," *In Proceeding of the 2019 World Wide Web Conference, ACM WWW*, 2019, pp. 417-426.

[36] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph Convolutional Matrix Completion," *arXiv preprint arXiv:1706.02263.*, 2017.

[37] M. Zhang, and Y. Chen, "Inductive Matrix Completion Based on Graph Neural Networks," *In Proceeding of the 8th International Conference on Learning Representations, ICLR*, 2020.

[38] X. Wang, R. Wang, C. Shi, G. Song, and Q. Li, "Multi-component graph convolutional collaborative filtering," *In Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, pp. 6267-6274.

[39] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural Graph Collaborative Filtering," *In Proceedings of the 42nd International Conference on Research and Development in Information Retrieval, ACM SIGIR* 2019, pp. 165-174.

[40] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," *In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 639-648.

[41] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie, "Self-supervised graph learning for recommendation," *In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 726-735.

[42] Z.-H. Zhou, and J. Feng, "Deep forest: Towards an alternative to deep neural networks," *In Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI* 2017, pp. 3553–3559.

[43] Z.-H. Zhou, and J. Feng, "Deep forest," *National Science Review,* vol. 6, no. 1, pp. 74-86, 2019.

[44] S. Rendle, W. Krichene, L. Zhang, and J. R. Anderson, "Neural Collaborative Filtering vs. Matrix Factorization Revisited," *In Proceedings of the 14th ACM Conference on Recommender Systems, RecSys*, 2020, pp. 240-248.

[45] Z.-H. Zhou, "Ensemble learning," *Encyclopedia of biometrics,* vol. 1, pp. 270-273, 2009.

[46] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *ACM Computing Surveys,* vol. 45, no. 1, pp. 1-40, 2012.

[47] N. I. Fisher, and P. K. Sen, "Probability Inequalities for Sums of Bounded Random Variables," *Publications of the American Statistical Association,* vol. 58, no. 301, pp. 13-30, 1963.

[48] N. Cesa-Bianchi, and G. Lugosi, "Prediction, learning, and games," *New York, NY, USA: Cambridge Univ. Press*, 2006.

[49] W. Rudin, "Principles of mathematical analysis," *McGraw-hill New York*, 1964.

[50] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Information Retrieval,* vol. 4, no. 2, pp. 133-151, 2001.

[51] D. Szklarczyk, A. L. Gable, D. Lyon, A. Junge, S. Wyder, J. Huerta-Cepas, M. Simonovic, N. T. Doncheva, J. H. Morris, and P. Bork, "STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets," *Nucleic Acids Research,* vol. 47, no. D1, pp. D607-D613, 2019.

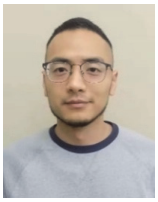
[52] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," *In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM*, 2018, pp. 333-341.

[53] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research,* vol. 7, no. Jan, pp. 1-30, 2006.

[54] D. Wu, X. Luo, M. Shang, Y. He, G. Wang and X. Wu, "A data-characteristic-aware latent factor model for Web services QoS prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 6, pp. 2525-2538, 2022.

[55] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Rethinking neural vs. matrix-factorization collaborative filtering: the theoretical perspectives," *In Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, pp.139:11514-11524.

[56] S. Duan, D. Wang, J. Ren, F. Lyu, Y. Zhang, H. Wu, and X. Shen., "Distributed Artificial Intelligence Empowered by End-Edge-Cloud Computing: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 591-624, 2023.



Di Wu (*Member*, IEEE) received his Ph.D. degree from the Chongqing Institute of Green and Intelligent Technology (CIGIT), Chinese Academy of Sciences (CAS), China in 2019 and then joined CIGIT, CAS, China. He is currently a Professor of the College of Computer and Information Science, Southwest University, Chongqing, China. He has over 60 publications, including 16 IEEE Transactions papers on T-SC, T-KDE, T-NNLS, T-SMC, etc., and several conference papers on ICDM, AAAI, WWW, IJCAI, etc. His research interests include machine learning and data mining. For more information please visit his homepage: https://wudi1989.github.io/Homepage/

Peng Zhang (Student Member, IEEE) received the B.S. degree in network engineering from the Shanxi Datong University, Datong, China, in 2018. He is currently pursuing the M.S. degree in computer technology at the Chongqing University of Posts and Telecommunications, Chongqing, China. He is a visiting student from July 2020 to present at the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences (CAS), Chongqing, China. His research interests include machine learning and data mining.

Yi He (*Member*, IEEE) received his Ph.D. degree from the University of Louisiana at Lafayette, USA in 2020 and then joined the Old Dominion University, USA where he is an Assistant Professor. His research outcomes have appeared in top venues, e.g., AAAI, IJCAI, WWW, ICDM, SDM, TKDE, TNNLS, to name a few. His research interests lie broadly in machine learning, data mining, and optimization theory. His homepage: https://www.lions.odu.edu/~y1he/index.html

Xin Luo (*Senior Member*, IEEE) received the B.S. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the Ph.D. degree in computer science from the Beihang University, Beijing, China, in 2011. He is currently a Professor of Data Science and Computational Intelligence with the College of Computer and Information Science, Southwest University, Chongqing, China. He has authored or coauthored over 200 papers (including over 120 IEEE Transactions papers) in the areas of his interests. His research interests focus on big data analysis and graph learning. Dr. Luo was the recipient of the Outstanding Associate Editor Award from IEEE/CAA JOURNAL OF AUTOMATICA SINICA in 2020. He is currently serving as a Deputy-Editor-in-Chief for IEEE/CAA JOURNAL OF AUTOMATICA SINICA, and an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. His Google page: https://scholar.google.com/citations?user=hyGlDs4AAAAJ&hl=zh-TW.